

ACTE ALE ORGANELOR DE SPECIALITATE ALE ADMINISTRAȚIEI PUBLICE CENTRALE

MINISTERUL EDUCAȚIEI ȘI CERCETĂRII

ORDIN

**pentru aprobarea programelor școlare asociate disciplinelor
prevăzute în Ordinul ministrului educației și cercetării nr. 4.350/2025
privind aprobarea planurilor-cadru pentru învățământul liceal cu frecvență zi*)**

În conformitate cu prevederile art. 88 alin. (2) și (10) din Legea învățământului preuniversitar nr. 198/2023, cu modificările și completările ulterioare,

având în vedere prevederile art. 4 din Hotărârea Guvernului nr. 251/2025 privind organizarea și funcționarea Centrului Național pentru Curriculum și Evaluare,

având în vedere Ordinul ministrului educației nr. 6.072/2023 privind aprobarea unor măsuri tranzitorii aplicabile la nivelul sistemului național de învățământ preuniversitar și superior, cu modificările ulterioare,

în baza Ordinului ministrului educației naționale nr. 3.593/2014 pentru aprobarea Metodologiei privind elaborarea și aprobarea curriculumului școlar — planuri-cadru de învățământ și programe școlare, cu modificările ulterioare,

ținând cont de Ordinul ministrului educației și cercetării nr. 4.350/2025 privind aprobarea planurilor-cadru pentru învățământul liceal cu frecvență zi,

având în vedere Referatul de aprobare nr. 2.832 din 16.12.2025,

în temeiul art. 13 alin. (3) din Hotărârea Guvernului nr. 731/2024 privind organizarea și funcționarea Ministerului Educației și Cercetării, cu modificările și completările ulterioare,

ministrul educației și cercetării emite prezentul ordin.

Art. 1. — Se aprobă Nota de prezentare generală a programelor școlare asociate disciplinelor prevăzute în Ordinul ministrului educației și cercetării nr. 4.350/2025 privind aprobarea planurilor-cadru pentru învățământul liceal cu frecvență zi, prevăzută în anexa nr. 1.

Art. 2. — Se aprobă Centralizatorul programelor școlare asociate disciplinelor prevăzute în Ordinul ministrului educației și cercetării nr. 4.350/2025 privind aprobarea planurilor-cadru pentru învățământul liceal cu frecvență zi, prevăzut în anexa nr. 2.

Art. 3. — Se aprobă programele școlare asociate disciplinelor prevăzute în Ordinul ministrului educației și cercetării nr. 4.350/2025 privind aprobarea planurilor-cadru pentru învățământul liceal cu frecvență zi, prevăzute în anexele nr. 3-89.

Art. 4. — Programele școlare prevăzute în anexele nr. 3-89 se aplică progresiv, începând cu clasa a IX-a a anului școlar 2026-2027.

Art. 5. — Programele școlare asociate fiecărui an de studiu sunt valabile pe o perioadă de 8 ani, începând cu anii școlari: 2026-2027 pentru clasa a IX-a, 2027-2028 pentru clasa a X-a, 2028-2029 pentru clasa a XI-a, respectiv 2029-2030 pentru clasa a XII-a.

Art. 6. — Anexele nr. 1-89 fac parte integrantă din prezentul ordin.

Art. 7. — Direcția generală echitate și performanță în învățământul preuniversitar, Direcția generală minorități și desegregare, Centrul Național pentru Curriculum și Evaluare, Centrul Național de Dezvoltare a Învățământului Tehnic și Profesional, inspectoratele școlare și unitățile de învățământ duc la îndeplinire prevederile prezentului ordin.

Art. 8. — Prezentul ordin se publică în Monitorul Oficial al României, Partea I.

Ministrul educației și cercetării,
Daniel-Ovidiu David

București, 19 decembrie 2025.
Nr. 6.930.

*) Ordinul nr. 6.930/2025 a fost publicat în Monitorul Oficial al României, Partea I, nr. 4 din 8 ianuarie 2026 și este reprodus și în acest număr bis.

Anexa nr. 1

NOTA DE PREZENTARE GENERALĂ
a programelor școlare asociate disciplinelor prevăzute în Ordinul ministrului
educației și cercetării nr. 4350/2025 privind aprobarea planurilor-cadru pentru
învățământul liceal cu frecvență zi

I. PREAMBUL

Curriculumul național reprezintă ansamblul coerent al documentelor care reglementează finalitățile, conținuturile, organizarea și evaluarea proceselor de predare-învățare-evaluare în învățământul preuniversitar și include planurile-cadru de învățământ, programele școlare și standardele naționale de evaluare.

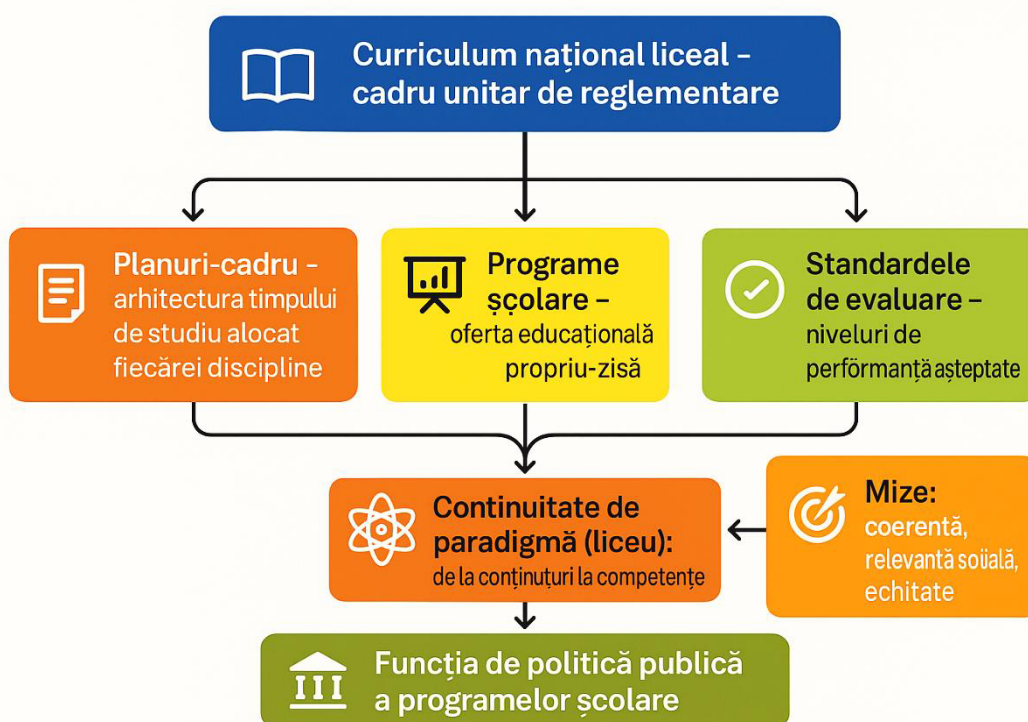


Figura 1 - Curriculumul național pentru învățământul liceal

Programele școlare pentru disciplinele obligatorii din învățământul liceal, prevăzute în planurile-cadru aprobate prin Ordinul ministrului educației și cercetării nr. 4350/2025, constituie documente curriculare reglatoare care:

- operaționalizează profilul de formare al absolventului;
- asigură continuitatea și progresia achizițiilor de la nivel gimnazial la nivel liceal;
- traduc în termeni didactici politicile publice în domeniul curriculumului școlar;
- fundamentează activitatea de predare-învățare-evaluare la nivelul clasei, al unității de învățământ și al sistemului educațional.

Prezenta Notă de fundamentare stabilește reperele conceptuale, normative și metodologice comune tuturor programelor școlare ale disciplinelor obligatorii din învățământul liceal - indiferent de tipul acestora (Trunchi comun - TC, Trunchi comun + Curriculum de specialitate - TC+CS, Curriculum de specialitate - CS) - și oferă cadrul de aplicare unitară a acestora.

Programele școlare nu sunt documente autonome, ci funcționează într-un sistem curricular integrat, în care:

- planurile-cadru stabilesc lista disciplinelor și arhitectura timpului de studiu;
- programele școlare definesc oferta educațională propriu-zisă;
- standardele naționale de evaluare vor preciza nivelurile de performanță așteptate.

Reperetele care au stat la baza procesului de elaborare a programelor școlare sunt:

- asigurarea coerenței interne între programele școlare ale diferitelor discipline;
- realizarea coerenței verticale între niveluri de învățământ;
- sprijinirea cadrelor didactice în utilizarea programelor școlare ca instrumente de proiectare didactică flexibilă și responsabilă;
- garantarea echității educaționale și a relevanței sociale și profesionale a ofertei curriculare liceale.

Elaborarea noilor programe școlare pentru învățământul liceal nu reprezintă o simplă actualizare de conținuturi, ci exprimă o *schimbare de paradigmă curriculară*, în deplin acord cu viziunea asupra educației consacrată prin Legea învățământului preuniversitar nr. 198/2023 și prin Nota de fundamentare a planurilor-cadru pentru liceu, aprobată prin OMEC nr. 4350/2025.

Această etapă este necesară deoarece:

- continuă modelul de proiectare curriculară aplicat deja în învățământul primar și gimnazial, care impune *tranziția de la un curriculum centrat preponderent pe conținuturi la un curriculum centrat pe competențe*, transferabile și utilizabile în contexte de viață reale, academice și profesionale;
- se afirmă explicit rolul formativ al liceului ca *etapă de sinteză, aprofundare și orientare*;
- se consolidează *profilul de formare al absolventului, ca element reglator al tuturor deciziilor curriculare*, de la arhitectura planurilor-cadru până la strategiile de evaluare;
- se recunoaște necesitatea unei *educații relevante social, economic și civic*, în contextul accelerării transformărilor tehnologice, digitale, ecologice și socio-economice;
- se instituie un *nou echilibru între cultura generală, formarea de specialitate și personalizarea parcursului educațional*, prin diferențierea clară între TC, TC+CS și CS.

În acest nou cadru, programele școlare devin *instrumente de politică publică educațională*, nu doar documente tehnice adresate procesului de predare-învățare-evaluare, având rolul de a transpune la nivelul practicii școlare:

- finalitățile majore ale învățământului liceal privind împlinirea personală, integrarea socială și inserția profesională;
- formarea gândirii critice, a capacității de decizie, a responsabilității civice și a adaptabilității;
- raportarea reflexivă și etică la cunoaștere, tehnologie și utilizarea informației.

În acord cu filosofia curriculară formulată în Nota de fundamentare a planurilor-cadru pentru învățământul liceal, noile programe școlare sunt concepute ca *documente deschise, flexibile și orientate spre sensul învățării*, care:

- sprijină dezvoltarea autonomiei în învățare a elevului;
- valorifică dimensiunea culturală, științifică, civică și profesională a educației;
- susțin formarea unei persoane reziliente, capabile să participe activ la viața socială și profesională;
- contracarează pseudoștiința, superficialitatea cognitivă și manipularea prin accentul pus pe rigoare, argumentare, reflecție și transfer.

Prin urmare, această etapă curriculară marchează *trecerea de la un curriculum al predării la un curriculum al formării*, de la logica disciplinei ca scop în sine la logica disciplinei coroborată cu logica didactică, *mijloace de construcție a profilului absolventului* și a capacității de a învăța pe tot parcursul vieții.

II. REPERE NORMATIVE ȘI STRATEGICE

II.1. Repere legislative

Elaborarea programelor școlare obligatorii pentru învățământul liceal s-a realizat în temeiul și în concordanță cu:

- **Legea învățământului preuniversitar nr. 198/2023**, cu modificările și completările ulterioare, care consacră abordarea curriculară centrată pe competențe și pe profilul de formare al absolventului;
- **Ordinul ministrului educației nr. 6731/2023** privind aprobarea Profilului de formare al absolventului;
- **Ordinul ministrului educației și cercetării nr. 4350/2025** privind aprobarea planurilor-cadru pentru învățământul liceal, forma cu frecvență zi;
- **Instrucțiunea nr. 8/2025** privind utilizarea timpului aflat la dispoziția cadrului didactic (25% din bugetul de timp), cu rol esențial în personalizarea învățării
- **Ordinul ministrului educației naționale nr. 3593/2014** pentru aprobarea Metodologiei privind elaborarea și aprobarea curriculumului școlar - planuri-cadru de învățământ și programe școlare

II.2. Repere strategice naționale, europene și internaționale

Programele școlare liceale sunt fundamentate și pe un ansamblu de documente de politici publice și strategii care vizează dezvoltarea durabilă, incluziunea, digitalizarea, educația pentru cetățenie și adaptarea educației la transformările economice și tehnologice contemporane, dintre care se disting:

- Strategia Națională pentru Dezvoltare Durabilă a României 2030;
- Strategia Națională privind educația pentru mediu și schimbările climatice 2023-2030;
- Strategia Națională de Educație Financiară 2024-2030;
- Strategia privind prevenirea violenței și promovarea stării de bine în mediul școlar;
- Recomandările Uniunii Europene privind competențele-cheie pentru învățarea pe tot parcursul vieții;
- Documentele OCDE și UNESCO privind competențele secolului XXI, învățarea socio-emoțională, educația pentru sustenabilitate și alfabetizarea digitală;
- Cadrul emergent de competențe pentru inteligența artificială și utilizarea etică a tehnologiilor digitale.

Aceste repere strategice au rol transversal și se regăsesc integrate în diferite componente ale programelor școlare (competențe generale, competențe specifice, conținuturi ale învățării, sugestii metodologice).

II.3. Repere curriculare și metodologice

Programele școlare din învățământul liceal au fost elaborate în acord cu o serie de documente elaborate la nivelul Centrului Național pentru Curriculum și Evaluare, precum:

- orientările pentru elaborarea programelor școlare liceale, care precizează principiile de selecție, structurare și corelare a competențelor, conținuturilor și activităților de învățare;
- sinteza documentelor strategice privind integrarea temelor transversale în programele școlare;
- tutorialul de elaborare a programelor școlare liceale, ca instrument metodologic unitar pentru asigurarea coerenței formale și conceptuale;
- recomandări privind curriculumul elaborat în sisteme educaționale de la nivel internațional, pentru identificarea unor bune practici .

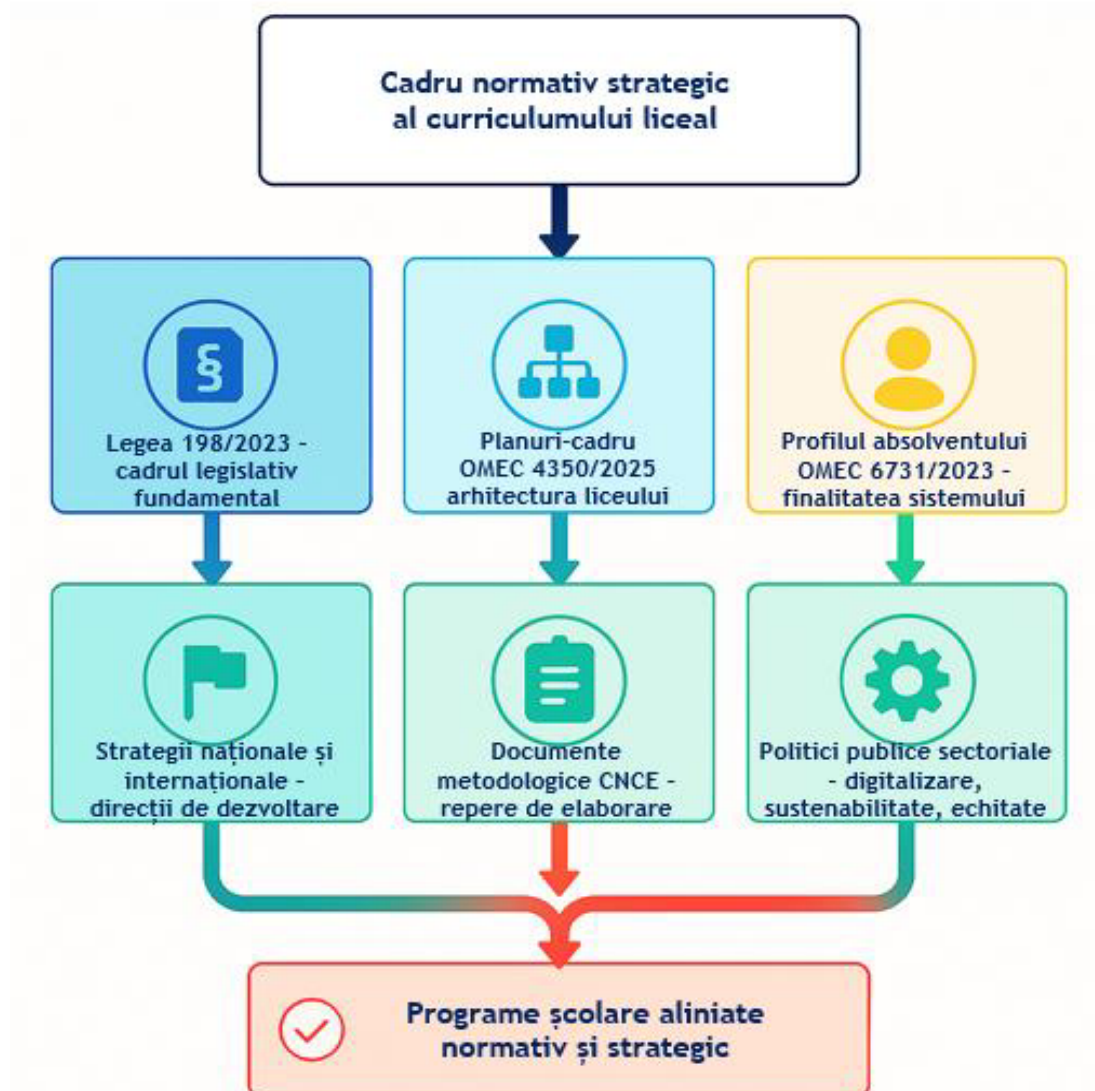


Figura 2 - Repere normative și strategice în elaborarea programelor școlare de liceu

III. PROFILUL DE FORMARE AL ABSOLVENTULUI - FUNCȚIE DE REGLARE PENTRU PROGRAME

Profilul de formare al absolventului, aprobat prin Ordinul ministrului educației nr. 6731/2023, reprezintă componenta fundamentală de referință pentru proiectarea și implementarea programelor școlare liceale.

Acesta este construit pe:

- *cele opt competențe-cheie europene*, asumate la nivel național, prin lege;
- *un set de nouă atribute prioritare* care descriu dimensiunile identitare, sociale, cognitive și valorice ale absolventului;
- *seturile de descriptori de nivel pentru învățământul liceal*, care definesc așteptările de final de ciclu și final al parcursului preuniversitar obligatoriu.

Astfel, programele școlare au rolul de a particulariza aceste competențe-cheie la nivelul fiecărei discipline de studiu, de a transforma atributele prioritare în comportamente formabile și observabile, respectiv de a structura progresia învățării pe parcursul educațional liceal.

În acest sens:

- *competențele generale* ale fiecărei discipline exprimă contribuția specifică a domeniului de studiu la profilul de formare al absolventului;
- *competențele specifice* reprezintă etape anuale de dezvoltare a competențelor generale;
- *conținuturile învățării* sunt selectate pentru a reprezenta un suport real în formarea și dezvoltarea competențelor;
- *activitățile de învățare și sugestiile metodologice* au rol de sprijin pentru profesori și sunt orientate prioritar spre formarea și transferul competențelor în contexte de viață socială, academice și profesionale.

IV. PARADIGMA COMPETENȚEI ÎN PROGRAMELE ȘCOLARE DIN ÎNVĂȚĂMÂNTUL LICEAL

Programele școlare pentru învățământul liceal sunt construite în mod unitar pe paradigma competenței, asumată explicit în legislația națională și în documentele de politici educaționale europene și internaționale. În acest cadru, competența este definită ca *ansamblu integrat și transferabil de cunoștințe, abilități și atitudini*, care permite elevului să acționeze eficient și responsabil în contexte variate de viață, învățare, muncă și cetățenie.

Aplicarea acestei paradigme în programele școlare liceale presupune următoarele repere fundamentale:

IV.1. Priorități în învățare

La nivel liceal, învățarea este orientată prioritar către:

- *formarea de capacități de analiză, reflecție, rezolvare de probleme și decizie;*
- *dezvoltarea autonomiei în învățare și a responsabilității personale;*
- *formarea disponibilității pentru învățarea pe tot parcursul vieții.*

În acest sens, cunoștințele nu constituie un scop în sine, ci sunt valorificate ca *resurse pentru dezvoltarea competențelor*, în relație directă cu situații autentice, relevante social, economic, civic și profesional.

IV.2. Structura sistemului de competențe în programele școlare liceale

În învățământul liceal, programele școlare operează coerent cu următoarele categorii de competențe:

- a) *competențele generale* - care exprimă contribuția domeniului de studiu/disciplinei la profilul de formare al absolventului, respectiv la dezvoltarea competențelor cheie;
- b) *competențele specifice*, derivate din competențele generale - care descriu achizițiile formabile și evaluabile pe durata unui an școlar.

Competențele specifice sunt formulate ca *etape progresive de dezvoltare a competențelor generale*, cu respectarea:

- coerenței verticale între clase;
- coerenței orizontale între discipline;
- nivelului de vârstă și de dezvoltare cognitivă a elevilor.

IV.3. Relația competențe - conținuturi - activități de învățare - evaluare

În noile programe școlare liceale, relațiile dintre componentele programei sunt stabilite în mod funcțional, astfel:

- *competențele determină selecția conținuturilor*, nu invers;
- *conținuturile sunt organizate în domenii de conținut* cu rol de suport pentru formarea competențelor;
- *exemplele de activități de învățare* acoperă complexitatea contextelor didactice și sunt orientate cu prioritate către participare activă, investigație, aplicație, reflecție și transfer;
- *evaluarea* este concepută ca proces continuu de reglare a învățării și de măsurare și apreciere a nivelului de dezvoltare a competențelor.

IV.4. Dimensiunea metacognitivă și transferul în viața reală

Un rol central în paradigma competenței îl are *metacogniția*, definită ca abilitatea elevului de a reflecta asupra propriului proces de învățare, de a-și autoregla strategiile de lucru și de a-și evalua progresul.

În acord cu orientările privind elaborarea programelor școlare liceale, metacogniția nu este tratată doar ca dimensiune psihopedagogică generală, ci ca *principiu transversal de proiectare curriculară*, care se reflectă explicit în:

- formularea competențelor specifice, prin integrarea proceselor de reflecție, autoevaluare, autoreglare a învățării;
- tipologia activităților de învățare, care favorizează:
 - explicarea propriilor demersuri de rezolvare;
 - analiza erorilor;
 - justificarea alegerilor făcute;
- orientarea evaluării către dimensiunea formativă și către feedback-ul constructiv.

Competențele formate la nivel liceal trebuie să fie *transferabile, funcționale și adaptabile*, nu limitate la contexte strict școlare. În acest cadru, programele școlare

liceale sunt concepute pentru a sprijini nu doar *ce învață elevul*, ci și *cum învață*, *de ce învață* și *cum își poate îmbunătăți propriul proces de învățare*, metacogniția având rolul de:

- a susține *autonomia elevului în învățare*, prin reflectarea asupra modului de învățare, respectiv formularea și revizuirea strategiilor personale de studiu;
- a facilita *transferul competențelor între discipline și în afara mediului școlar*, în situații de viață cotidiană, academice și profesionale;
- a consolida *capacitatea de adaptare la sarcini noi, contexte noi și situații neprevăzute*.

IV.5. Particularități ale competențelor în funcție de tipul de programă

În toate dintre următoarele situații se are în vedere atât o unitate de viziune asupra competenței, ca achiziție integrată și transferabilă, dar aceasta capătă valențe diferențiate, astfel:

- a) *pentru disciplinele de trunchi comun (TC)*, competențele vizează formarea unor achiziții de bază, comune tuturor elevilor;
- b) *pentru disciplinele exclusiv de curriculum de specialitate (CS)*, competențele sunt orientate prioritar spre dezvoltarea de cunoștințe, abilități și atitudini în raport cu cerințele domeniului de specializare și, după caz, cu nivelul de calificare;
- c) *pentru disciplinele de tip TC+CS*, competențele generale sunt comune tuturor elevilor, iar competențele specifice și conținuturile asociate curriculumului de specialitate reflectă aprofundarea și diferențierea în funcție de profil și specializare.

IV.6. Aspecte specifice în învățământul tehnologic

În cazul disciplinelor de specialitate și modulelor de pregătire din liceul tehnologic, paradigma competenței este corelată cu *modelul rezultatelor învățării* specifice calificărilor profesionale, definite în termeni de:

- cunoștințe;
- abilități;
- autonomie și responsabilitate.

Programele școlare pentru aceste domenii sunt corelate cu *standardele de pregătire profesională (SPP)*, asigurând compatibilitatea cu Cadrul European al Calificărilor și cu cerințele pieței muncii.

IV.7. Rolurile programelor școlare

Conform modelului curricular centrat pe competențe, aplicarea programelor școlare implică nu doar respectarea conținuturilor propuse, ci mai ales asumarea *finalităților formative exprimate prin competențe*. Astfel, programele școlare nu au rolul de simple liste de conținuturi, ci devin:

- instrumente de proiectare a învățării;
- repere pentru personalizarea demersului didactic;
- cadre de referință pentru evaluarea formativă și sumativă;
- documente fundamentale pentru asigurarea calității educației.

V. PRINCIPII CURRICULARE CARE AU GHIDAT ELABORAREA PROGRAMELOR ȘCOLARE PENTRU ÎNVĂȚĂMÂNTUL LICEAL

Elaborarea programelor școlare pentru disciplinele/modulele de pregătire obligatorii pentru învățământul liceal s-a realizat pe baza unui set coerent de *principii curriculare fundamentale*, cu rol de asigurare a calității, coerenței și relevanței curriculare. Aceste principii sunt operaționalizate în toate componentele programelor școlare - competențe, conținuturi, activități de învățare, sugestii metodologice și evaluare.

V.1. Principiul coerenței curriculare

Coerența constituie o condiție esențială pentru *unitatea și stabilitatea curriculumului național* și este asigurată la două niveluri:

- a) *coerența verticală*, între diferitele niveluri și ani de studiu;
- b) *coerența orizontală (intra-, inter-)* între domeniile proprii unei discipline, între disciplinele din aceeași arie curriculară și dintre arii diferite, prin evitarea suprapunerilor nejustificate și prin susținerea abordărilor interdisciplinare.

Asigurarea coerenței verticale între ciclul gimnazial și ciclul liceal se realizează prin:

- corelarea competențelor generale liceale cu cele formate la nivel gimnazial;
- asigurarea **progresiei nivelului de complexitate cognitivă**, de la achiziții fundamentale la operare teoretică și până la analiză critică, modelare și decizie;
- evitarea suprapunerilor inutile de conținuturi și a reluărilor nefuncționale;
- reconstrucția conținuturilor din perspectivă **formativă și aplicativă**, specifică vârstei elevilor de liceu.

În acest sens, programele școlare pentru învățământul liceal nu reprezintă o continuare și dezvoltare tematică a celor gimnaziale, ci o *etapă de sinteză, conceptualizare, problematizare și orientare către viața adultă, studiile superioare și carieră*. Coerența verticală devine astfel o *condiție de calitate curriculară*, care asigură:

- fluența parcursului educațional al elevului;
- reducerea fragmentării achizițiilor;
- creșterea eficienței formării competențelor pe termen lung.

V.2. Principiul selecției, al relevanței și al ierarhizării culturale

Programele școlare reflectă un decupaj riguros și actualizat al domeniilor de cunoaștere, realizat în raport cu:

- evoluțiile științifice, tehnologice, economice și culturale;
- elementele de noutate în domeniile cu relevanță educațională (psihologia învățării, abordări inovative în predare, priorități în evaluare etc.)
- cerințele societății contemporane;
- profilul de formare al absolventului.

Conținuturile învățării au fost selectate astfel încât să:

- evite supraîncărcarea informativă;
- favorizeze înțelegerea conceptelor fundamentale;
- permită transferul achizițiilor în contexte variate.

Ierarhizarea culturală asigură *cunoașterea* specifică fiecărei discipline, în paralel cu deschiderea către problematici contemporane.

V.3. Principiul centrării pe elev

Din perspectiva acestui principiu, programele școlare urmăresc să promoveze:

- o învățare activă, în care elevul este direct implicat în propria învățare, în rezolvarea de probleme din viața reală;
- o învățare contextuală, în care noile achiziții se construiesc pornind de la baza preexistentă de cunoaștere;
- o învățare socială, care sprijină cooperarea și colaborarea între elevi;
- o învățare responsabilă, cu accent pe pregătirea elevilor de a-și formula obiective și priorități în învățare.

Punctul central al procesului didactic se deplasează de la predare la *învățare activă, participativă și reflexivă*, în care elevul are rol de subiect (agent) al propriei formări.

V.4. Principiul calității măsurabile și al evaluării orientate spre competențe

Programele școlare sunt elaborate astfel încât să permită:

- formularea unor competențe clare, observabile și evaluabile;
- corelarea acestora cu standardele naționale de evaluare;
- utilizarea evaluării nu doar ca instrument de certificare, ci ca mijloc de reglare a învățării.

Evaluarea este concepută integrat în procesul didactic, cu accent pe:

- feedback formativ;
- autoevaluare;
- progres și dezvoltare.

V.5. Principiul corelării cu particularitățile de vârstă ale elevilor

Structura competențelor și selecția conținuturilor sunt adaptate la:

- particularitățile de dezvoltare cognitivă, emoțională și socială ale elevilor de liceu;
- ritmurile diferite de învățare;
- interesele și aspirațiile personale și nevoile de orientare școlară și profesională .

Numărul competențelor generale și al competențelor specifice este corelat cu bugetul de timp alocat fiecărei discipline prin planurile-cadru, pentru a asigura realizabilitatea și eficiența formării.

V.6. Principiul subsidiarității, al flexibilității și al parcursului individual

Programele școlare permit:

- parcurgerea programei în raport cu logica internă a fiecărei discipline;
- adaptarea demersului didactic la nivelul clasei și al elevilor;
- personalizarea învățării;
- utilizarea diferențiată a resurselor educaționale.

Un rol central îl are utilizarea *timpului aflat la dispoziția cadrului didactic (25%)*, ca rezervă de ore care nu au fost luate în considerare în raport cu încărcarea programelor, dar care au fost avute în vedere pentru posibile activități de completare, consolidare, remediere, aprofundare, performanță și/sau extindere și transfer în contexte reale. Astfel, se asigură echilibrul dintre obligațiile ce decurg din programa școlară și autonomia profesională a cadrului didactic.

V.7. Principiul egalității de șanse și al educației incluzive

Programele școlare pentru învățământul liceal:

- propun **același set de competențe specifice pentru toți elevii**, în cadrul aceleiași discipline și aceluiasi tip de parcurs educațional;
- asigură acces echitabil la achizițiile fundamentale de învățare;
- sunt deschise spre adaptări pedagogice pentru elevii cu ritmuri diferite de învățare sau cu cerințe educaționale speciale.

Prin aceasta, curriculumul liceal contribuie la prevenirea excluziunii educaționale, la reducerea abandonului școlar și la valorizarea diversității.

V.8. Principiul funcționalității

Funcționalitatea internă a programelor școlare este asigurată prin:

- corelarea competențelor specifice cu exemplele de activități de învățare;
- corelarea competențelor specifice cu conținuturile învățării;
- corelarea competențelor cu sugestiile metodologice.

Aceste niveluri de corelare garantează faptul că fiecare programă are coerență internă, este orientată efectiv spre formarea competențelor și este aplicabilă în viața de zi cu zi și în raport cu viitoare contexte profesionale.

V.9. Principiul racordării la social, la viața reală și la piața muncii

Programele școlare sunt fundamentate pe documente actuale de politici publice, pe cercetări educaționale și pe analiza dinamicii sociale și economice, urmărind:

- relevanța învățării pentru viața cotidiană;
- dezvoltarea competențelor necesare inserției profesionale;
- stimularea spiritului civic și a responsabilității sociale;
- adaptarea la transformările generate de digitalizare, globalizare și tranziția ecologică.

V.10. Principiul educației pentru valori și etică

Programele școlare pentru învățământul liceal promovează explicit:

- respectul pentru demnitatea umană;

- gândirea critică;
- dialogul și cooperarea;
- responsabilitatea față de sine, față de ceilalți și față de mediu;
- integritatea intelectuală și etica utilizării informației.

Dimensiunea axiologică - sistemul de valori, atitudini, principii etice și repere morale - este considerată *parte integrantă a competențelor formulate la nivelul tuturor disciplinelor de studiu*, nu doar a unor discipline specifice. **Integrarea acestei dimensiuni este esențială pentru formarea elevului ca:**

- persoană autonomă;
- actor social;
- viitor profesionist responsabil.

VI. TIPOLOGIA PROGRAMELOR ȘCOLARE PENTRU ÎNVĂȚĂMÂNTUL LICEAL

Programele școlare liceale pentru disciplinele obligatorii se diferențiază după statutul curricular (Trunchi comun - TC, Curriculum de specialitate - CS, respectiv TC+CS), dar au o structură comună, care asigură coerența curriculumului național și facilitează aplicarea unitară la nivelul sistemului.

VI.1. Tipologia programelor școlare liceale (TC, TC+CS, CS)

În concordanță cu planurile-cadru aprobate prin OMEC nr. 4350/2025 și cu orientările metodologice pentru elaborarea programelor școlare liceale, programele se încadrează în trei tipuri principale:

a) Programe școlare pentru discipline de trunchi comun (TC), cu adresabilitate către toți elevii de liceu, indiferent de filieră, profil, specializare/calificare profesională, în acord cu specificațiile planurilor-cadru. Acestea au ca finalitate formarea unui set de competențe generale de bază, comune și obligatorii, cu rol de fundament cultural, științific, civic și personal, asigurând *nucleul comun al formării* la nivel liceal, în legătură directă cu profilul de formare al absolventului.

Implicații curriculare:

- competențele generale și specifice sunt formulate astfel încât să fie relevante pentru întreaga cohortă de elevi;
- conținuturile vizează achiziții durabile și transferabile;
- activitățile de învățare susțin diferențierea.

b) Programe școlare pentru discipline de curriculum de specialitate (CS), cu adresabilitate către toți elevii care urmează o anumită filieră/profil/specializare sau calificare profesională, conform planurilor-cadru. Acestea au ca finalitate formarea de competențe de aprofundare și specializare, , susținând identitatea specifică a traseelor educaționale și pregătind elevii pentru studii universitare în domenii conexe și/sau pentru inserție profesională în domenii tehnice sau vocaționale.

Implicații curriculare:

- competențele generale sunt specifice domeniului (de exemplu: profil umanist, profil tehnologic, profil artistic etc.) și specializării/calificării;
- conținuturile și activitățile de învățare sunt calibrate pe nevoile academice/profesionale ale profilului și specializării/calificării;

c) *Programe școlare pentru discipline de tip TC+CS*, cu adresabilitate către elevii din anumite profiluri/specializări sau calificări profesionale care au în CS discipline ce se studiază și în TC. Aceste programe școlare includ competențele de bază specificate în TC și le completează cu competențe de aprofundare corespundente cu profilurile/specializările cărora li se adresează.

Implicații curriculare:

- se evită dublarea sau fragmentarea celor două programe adresate aceleiași discipline, printr-o partajare clară sau integrare explicită a celor două componente (TC și CS);
- după caz, se explicitează, în cadrul programei, care competențe și conținuturi sunt de TC și care aparțin CS.

d) Particularități pentru învățământul liceal tehnologic

Pentru învățământul liceal tehnologic:

- disciplinele de cultură generală din TC urmează structura comună descrisă la subsecțiunea VI.2;
- pentru modulele de pregătire de specialitate, programele pot fi structurate în raport cu modelul bazat pe rezultate ale învățării și standardele de pregătire profesională, conform cadrului specific învățământului tehnologic.

În aceste situații, structura poate include elemente specifice (unități de rezultate ale învățării, criterii de performanță, condiții de realizare), cu respectarea însă a principiilor generale ale prezentei Note.

VII. STRUCTURA PROGRAMELOR ȘCOLARE PENTRU ÎNVĂȚĂMÂNTUL LICEAL

Indiferent de tipul de programă (TC, CS sau TC+CS), programele școlare pentru disciplinele din învățământul liceal au o structură comună.

VII.1. Secțiunile programelor școlare

Principalele secțiuni ale programelor școlare sunt:

1. *Nota de prezentare;*
2. *Competențe generale;*
3. *Competențe specifice (pe ani de studiu);*
4. *Exemple de activități de învățare;*
5. *Domenii de conținut și conținuturi ale învățării;*
6. *Sugestii metodologice.*

Suplimentar, după caz, programele școlare pot include anexe care au caracter orientativ, fiind menite să sprijine aplicarea programei, fără a adăuga noi obligații. Anexele pot face referire la: glosar de termeni, matrice de corelare între competențe și conținuturi, exemple de strategii didactice sau de instrumente de evaluare etc.

VII.2. Rolul fiecărei secțiuni și relația dintre acestea

1. Nota de prezentare

- precizează **statutul disciplinei** (TC/CS/TC+CS) și timpul de studiu alocat
- formulează finalitățile și relevanța disciplinei în raport cu profilul de formare
- indică, după caz, elementele de continuitate sau de relație, respectiv de noutate față de nivelul gimnazial
- oferă orientările generale de lectură și aplicare a programei
-

2. Competențe generale

- exprimă contribuția majoră a disciplinei la profilul de formare al absolventului
- sunt formulate la nivelul întregului parcurs liceal al disciplinei
- reprezintă baza de derivare a competențelor specifice
- sunt *obligatorii, nereductibile și neselectabile* de către profesor

3. Competențe specifice

- sunt derivate din competențele generale și descriu *achizițiile intermediare* prin care se formează acestea la nivelul fiecărui an de studiu
- sunt formulate în termeni observabili și evaluabili
- integrează dimensiuni cognitive, procedurale și axiologice
- sunt *obligatorii*; profesorul nu le modifică, ci le *operaționalizează* în unități de învățare și activități concrete

4. Exemple de activități de învățare

- ilustrează *moduri posibile* de a proiecta situații de învățare prin care competențele specifice pot fi formate, utilizând conținuturile propuse
- sprijină profesorii în proiectarea didactică, în selectarea strategiilor de predare-învățare-evaluare
- *reprezintă elemente orientative*, nu liste exhaustive și nu prescripții de metodă, profesorul având libertatea de:
 - a adapta activitățile propuse
 - a concepe activități noi
 - a personaliza activitățile pentru diverse categorii de elevi
 - a completa cadrul de exemplificare cu activități destinate orelor la dispoziția sa (25%)
- relaționare internă (în cadrul programei școlare): *fac vizibilă legătura concretă* dintre competențe specifice și conținuturi

5. Domenii de conținut și conținuturi ale învățării

- domeniile de conținut reflectă *structura internă a disciplinei* (marile capitole/axe tematice)
- conținuturile sunt *selectate în raport direct cu competențele specifice*, ca suport pentru formarea acestora
- sunt *obligatorii*
- profesorul are un grad de libertate în:
 -

- parcurgerea domeniilor/conținuturilor în ordinea stabilită de acesta, în raport cu logica internă a disciplinei, nu în ordinea enumerării lor în programa școlară
- organizarea acestora în unități de învățare
- alocarea de timp în planificare, în acord cu nevoile de învățare ale elevilor

6. Sugestii metodologice

- oferă repere privind:
 - scopul, înțelesurile, oportunitățile și limitele asociate domeniilor de conținut/conținuturilor, mai ales în situații de noutate a acestora
 - strategii didactice recomandate (fără caracter limitativ)
 - utilizarea resurselor (inclusiv digitale)
 - modalități de diferențiere și de personalizare
 - integrarea temelor transversale
 - utilizarea timpului de 25% la dispoziția cadrului didactic, ca suport suplimentar în atingerea finalităților propuse prin programa școlară
 - orientări privind evaluarea (în corelare cu viitoarele standarde)
 - sunt *orientative*, nu normative la nivel de metodă
 - valorifică autonomia profesională a cadrului didactic, oferind în același timp repere de calitate

VII.3. Elemente obligatorii, elemente cu grade de libertate, elemente orientative

Sintetizând, programele școlare au elemente:

- *obligatorii, fără opțiuni de modificare:*
 - statutul disciplinei (TC/CS/TC+CS, număr de ore), conform planurilor-cadru;
 - competențele generale;
 - competențele specifice;
 - domeniile de conținut (ca structură macro);
 - poziționarea generală a disciplinei în raport cu profilul de formare;
- *obligatorii, dar cu grade de libertate în aplicare:*
 - conținuturile (tematicile de studiu), în limitele stabilite de programă, dar cu libertate de:
 - ordonare a lor în planificarea calendaristică;
 - pondere acordată;
 - exemplificări, selecții din liste date în programă;
 - utilizarea timpului de 25% la dispoziția cadrului didactic, în completarea cadrului oferit de programă;
- *orientative:*
 - exemplele de activități de învățare;
 - sugestiile metodologice;
 - după caz, anexele (matrice, instrumente-model etc.).

Profesorul are *obligația de a forma toate competențele și de a parcurge domeniile și conținuturile prevăzute în programă*, dar beneficiază de *autonomie profesională* în:

- *planificarea și modul de organizare a unităților de învățare;*
- *selecția activităților și a strategiilor didactice;*
- *adaptarea demersului la contextul clasei, la nevoile elevilor.*

VIII. TEME TRANSVERSALE MAJORE ȘI MODUL LOR DE INTEGRARE ÎN PROGRAMELE ȘCOLARE LICEALE

Programele școlare pentru învățământul liceal integrează, în principal în segmentul trunchiului comun, un set de *teme transversale majore*, asumate la nivel național, european și internațional prin documente de politici publice și strategii sectoriale. Aceste teme nu constituie discipline distincte (decât în situații punctuale prevăzute de planurile-cadru), ci sunt *integrate funcțional în competențe, conținuturi, activități de învățare și sugestii metodologice*, în funcție de specificul fiecărei discipline.

Temele transversale nu reprezintă o simplă „listă de intenții”, ci un cadru operațional de legătură între curriculumul școlar, pe de o parte, și problemele și prioritățile din plan personal, social, profesional, natural, pe de altă parte.

Integrarea lor urmărește formarea unei *viziuni unitare asupra lumii contemporane, dezvoltarea gândirii sistemice, a responsabilității civice și a capacității de decizie informată*.

Integrarea temelor transversale în programele școlare respectă următoarele principii:

- nu conduce la *supraîncărcarea conținuturilor*, ci la *recontextualizarea și valorificarea acestora*;
- nu necesită noi discipline, ci *sensuri formative suplimentare* pentru competențele existente;
- este realizată *diferențiat*, în funcție de: natura disciplinei, tipul de programă (TC, CS, TC+CS), nivelul de vârstă și profilul elevilor;
- este sprijinită prin: teme specifice, exemple de activități de învățare, sugestii metodologice, utilizarea flexibilă a timpului la dispoziția cadrului didactic.

VIII.1. Educația pentru dezvoltare durabilă și economia circulară

Educația pentru dezvoltare durabilă (EDD) reprezintă un reper transversal fundamental al programelor liceale, fiind integrată din perspectiva:

- dimensiunii *ecologice* (protecția mediului, schimbări climatice, biodiversitate), cu accent pe *competențele verzi*;
- dimensiunii *economice* (utilizarea responsabilă a resurselor, producție și consum sustenabile, eficiență);
- dimensiunii *sociale* (echitate, coeziune socială, responsabilitate față de generațiile viitoare).

Economia circulară este tratată ca *aplicație operațională a EDD*, vizând:

- reducerea risipei;
- reutilizarea și reciclarea;
- proiectarea durabilă a produselor și serviciilor;
- comportamentele individuale și colective responsabile.

Integrarea se realizează:

- explicit, în competențele și conținuturile disciplinelor de științe, geografice, tehnologice, economice și socio-umane;

- implicit, la diferite discipline de studiu, prin exemplele de activități de învățare de tip proiect, investigație, studiu de caz și rezolvare de probleme.

VIII.2. Educația digitală și utilizarea responsabilă a tehnologiei, inclusiv a inteligenței artificiale

Educația digitală este tratată transversal, dincolo de disciplinele Informatică și Tehnologia informației și a comunicațiilor (TIC), vizând:

- dezvoltarea competenței digitale ca *instrument de gândire, comunicare și creație*;
- utilizarea tehnologiei în *procesul de învățare, documentare, modelare și evaluare*;
- formarea gândirii critice față de informația digitală, platforme, algoritmi.

Utilizarea inteligenței artificiale este abordată dintr-o dublă perspectivă:

- *funcțională* - ca instrument de sprijin pentru învățare, creație, analiză;
- *etică și critică* - privind impactul social, riscurile, limitele, responsabilitatea utilizării.

Integrarea presupune:

- dezvoltarea capacității de a decide *când, cum și cu ce scop este adecvată utilizarea tehnologiei*;
- protecția datelor, respectarea proprietății intelectuale și a demnității umane, prin elemente de securitate cibernetică și siguranță pe internet;
- utilizarea tehnologiilor digitale ca instrumente de studiu și învățare.

VIII.3. Învățarea socio-emoțională (SEL), starea de bine și cultura școlară pozitivă

Învățarea socio-emoțională este integrată transversal în programele școlare prin:

- dezvoltarea competențelor personale, sociale și civice;
- cultivarea empatiei, respectului reciproc, comunicării asertive;
- prevenirea violenței, discriminării, bullying-ului și cyberbullying-ului;
- formarea capacității de gestionare a emoțiilor, stresului și conflictelor.

Această temă este prezentă **funcțional în toate disciplinele**, prin tipul de interacțiune educațională promovată, dar este corelată prioritar cu:

- disciplinele de consiliere și orientare;
- disciplinele de limbă și comunicare;
- educația fizică, disciplinele artistice și socio-umane.

VIII.4. Educația financiară și antreprenorială

Educația financiară și antreprenorială este integrată ca temă transversală, cu rol în:

- dezvoltarea gândirii economice de bază;
- formarea capacității de a gestiona resursele personale;
- înțelegerea mecanismelor economice fundamentale;
- asumarea inițiativei, a riscului calculat și a responsabilității decizionale.

Integrarea se realizează atât explicit, în disciplinele dedicate (economie, educație antreprenorială), cât și implicit, în matematică, științe, tehnologii, studii sociale, prin aplicații, proiecte și situații-problemă autentice.

VIII.5. Educația pentru cetățenie democratică, juridică și media

Programele școlare pentru învățământul liceal integrează transversal:

- educația pentru drepturile omului, inclusiv din perspectiva egalității de șanse între femei și bărbați;
- înțelegerea mecanismelor democratice, pentru o expresie a cetățeniei democratice, inclusiv europene;
- cultura juridică de bază;
- educația media și raportarea critică la mass-media și rețelele sociale.

Această integrare urmărește:

- formarea capacității de participare civică informată;
- exersarea gândirii critice;
- asumarea responsabilității sociale;
- combaterea dezinformării și manipulării.

VIII.6. Educația pentru sănătate și stil de viață echilibrat

Tematica sănătății este integrată transversal, vizând:

- sănătatea fizică, mintală și emoțională;
- prevenția comportamentelor de risc, inclusiv cele legate de educația sexuală;
- stilul de viață activ și echilibrat;
- igiena, nutriția, echilibrul muncă-odihnă.

Integrarea se realizează explicit, prin disciplinele de specialitate, respectiv implicit, prin modul de proiectare al activităților de învățare și al mediului educațional.

VIII.7. Integrarea temelor transversale - principii de aplicare

Integrarea temelor transversale în programele școlare respectă următoarele principii:

- nu conduce la *supraîncărcarea conținuturilor*, ci la *recontextualizarea și valorificarea acestora*;
- nu necesită noi discipline , ci *sensuri formative suplimentare* pentru competențele existente;
- este realizată *diferențiat*, în funcție de:
 - o natura disciplinei;
 - o tipul de programă (TC, CS, TC+CS);
 - o nivelul de vârstă și profilul elevilor;
- este sprijinită prin:
 - o teme specifice;
 - o exemple de activități de învățare;
 - o sugestii metodologice;
 - o utilizarea flexibilă a timpului la dispoziția cadrului didactic.

IX. ROLUL CADRULUI DIDACTIC ȘI AUTONOMIA PROFESIONALĂ ÎN APLICAREA PROGRAMELOR ȘCOLARE LICEALE

Aplicarea programelor școlare liceale se realizează cu responsabilitatea directă a cadrului didactic, în condițiile respectării caracterului obligatoriu al acestora și ale principiilor autonomiei profesionale, consacrate prin legislația în vigoare. Programele școlare stabilesc *ce finalități trebuie atinse și ce achiziții sunt obligatorii pentru elevi, în timp ce cadrul didactic are libertatea profesională de a decide cum sunt realizate aceste finalități, în funcție de contextul educațional concret.*

IX.1. Statutul programelor școlare în raport cu proiectarea didactică

Programele școlare constituie documente curriculare normative, cu caracter obligatoriu pentru toate unitățile de învățământ și pentru toate cadrele didactice care predau disciplina respectivă. Fiecare programă conține elemente cu caracter de obligativitate, cu grade de libertate și orientative (detaliat în capitolul anterior).

Proiectarea didactică anuală și pe unități de învățare este *instrumentul prin care cadrul didactic operaționalizează programa școlară.*

IX.2. Autonomia profesională și responsabilitatea cadrului didactic

Autonomia profesională a cadrului didactic se exercită în limitele stabilite de cadrul normativ național, programele școlare pentru disciplinele obligatorii, precum și prin raportare la principiile echității și ale interesului superior al elevului.

Această autonomie presupune:

- libertatea de a selecta și adapta metodele de predare-învățare-evaluare;
- capacitatea de a construi situații de învățare relevante și diferențiate;
- dreptul și obligația de a utiliza resurse educaționale variate;
- responsabilitatea pentru calitatea actului educațional și pentru progresul elevilor.

Autonomia profesională este puternic legată de *responsabilitatea pedagogică*, care implică:

- asumarea finalităților programelor școlare;
- respectarea principiilor de deontologie;
- fundamentarea deciziilor didactice pe argumente științifice, pedagogice și etice;
- raportarea constantă la achizițiile anterioare, stilurile și ritmurile de învățare, precum și la nevoile și interesele reale ale elevilor.

IX.3. Utilizarea timpului aflat la dispoziția cadrului didactic (25%)

În aplicarea programelor școlare, cadrul didactic dispune de *25% din bugetul de timp alocat disciplinei*, conform Instrucțiunii nr. 8/2025, cu următoarele destinații principale:

- remediere și sprijin pentru elevii cu dificultăți de învățare;
- consolidare și aprofundare a achizițiilor;
- extindere și transfer al competențelor în contexte variate;
- activități de performanță;
- proiecte interdisciplinare și aplicații practice.

Aceste ore:

- reprezintă timp suplimentar față de cel la care s-a raportat programa școlară, dar este parte integrantă a bugetului de timp al disciplinei;
- sunt planificate explicit în documentele de proiectare didactică;
- se utilizează în baza unui diagnostic inițial și periodic al achizițiilor elevilor;
- contribuie la personalizarea parcursului educațional și la creșterea calității învățării.

IX.4. Personalizarea învățării

Personalizarea nu afectează setul comun de competențe obligatorii ale disciplinei, ci reprezintă modalități diferite de atingere a acestora și finalități. Astfel, programele școlare pentru învățământul liceal sunt concepute astfel încât să permită:

- adaptarea demersului didactic la niveluri diferite de pregătire;
- varierea ritmurilor de învățare;
- utilizarea strategiilor de grupare flexibilă a elevilor în cadrul clasei sau la nivelul unității de învățământ.

În acest sens, cadrul didactic poate proiecta activități diferențiate și sarcini de lucru cu grade diferite de complexitate pentru a genera trasee de învățare personalizate, în acord cu potențialul elevilor.

IX.5. Rolul cadrului didactic în integrarea temelor transversale

Cadrul didactic are rol esențial în:

- integrarea temelor transversale în activitățile de învățare;
- valorificarea contextelor reale și interdisciplinare;
- articularea conținuturilor propriei discipline cu problematici sociale, tehnologice, economice și etice actuale.

Această integrare se realizează prin:

- selecția situațiilor de învățare;
- proiecte interdisciplinare;
- utilizarea timpului la dispoziția cadrului didactic;
- colaborarea cu alți profesori și cu parteneri educaționali.

IX.6. Corelarea cu evaluarea formativă

În aplicarea programelor școlare, cadrul didactic are responsabilitatea de a utiliza *evaluarea formativă* ca instrument esențial de:

- monitorizare continuă a progresului elevilor, în raport cu *competențele specifice*;
- identificarea dificultăților și a lacunelor de parcurs, concomitent cu nevoile de sprijin;
- reglare și adaptare a demersului didactic.

Evaluarea formativă se realizează printr-o varietate de metode și instrumente (evaluări scrise, evaluări orale, observație sistematică, portofoliu, proiecte, investigații, autoevaluări și evaluări între egali etc.) și:

- este direct corelată cu activitățile de învățare;
- oferă *feedback descriptiv, orientat spre progres*, nu doar calificativ sau notă;
- pune accent pe procesul de învățare (cum învață elevul, ce dificultăți și provocări întâmpină, ce nevoi de suport are), nu exclusiv pe rezultatul învățării.

Programele școlare oferă repere orientative privind evaluarea (în special în secțiunea de sugestii metodologice), lăsând cadrului didactic libertatea profesională de a alege formele concrete de evaluare formativă, cu condiția respectării competențelor vizate și a principiilor obiectivității, echității și transparenței.

IX.7. Valorificarea tehnologiilor digitale și a inteligenței artificiale de către cadrul didactic pentru facilitarea învățării

În aplicarea programelor școlare liceale, cadrul didactic valorifică tehnologiile digitale și instrumentele bazate pe inteligență artificială ca resurse educaționale care:

- sprijină accesul elevilor la surse variate de informare;
- promovează învățarea activă, exploratorie și colaborativă;
- permite personalizarea parcursurilor de învățare;
- oferă oportunități de evaluare formativă și feedback rapid.

Utilizarea tehnologiei și a aplicațiilor bazate pe inteligență artificială se realizează în mod critic-reflexiv și responsabil, în acord cu finalitățile educaționale stabilite prin programele școlare, cu principiile eticii, protecției datelor și proprietății intelectuale, precum și cu vârsta elevilor și specificul disciplinei.

În acest sens, cadrul didactic:

- selectează și utilizează instrumente digitale și soluții bazate pe inteligență artificială numai în măsura în care acestea sprijină formarea competențelor prevăzute în programă;
- integrează tehnologia ca mijloc de facilitare a învățării, nu ca substitut al gândirii autonome, al raționamentului și al efortului intelectual propriu al elevului;
- ghidează elevii în utilizarea critică a rezultatelor generate de sisteme digitale și de inteligență artificială, învățându-i să verifice acuratețea, relevanța și limitele acestora;
- atrage atenția asupra riscurilor utilizării defectuoase a tehnologiilor digitale sau a suprautilizării lor;
- creează contexte de învățare în care tehnologia favorizează creativitatea, colaborarea, modelarea, simularea și rezolvarea de probleme.

Programele școlare susțin utilizarea tehnologiilor și a inteligenței artificiale în măsura în care acestea contribuie la:

- creșterea calității actului educațional;
- dezvoltarea competenței digitale a elevilor;
- formarea unei relații responsabile între elev, cunoaștere și tehnologie.

Responsabilitatea deciziei privind oportunitatea și modul de utilizare a tehnologiilor și a aplicațiilor bazate pe inteligență artificială revine cadrului didactic, în exercitarea autonomiei sale profesionale, în interesul superior al elevului și al calității învățării.

IX.8. Responsabilități ale cadrului didactic în programele naționale „Săptămâna verde” și „Școala altfel”

În acord cu reglementările privind organizarea programelor naționale „Săptămâna verde” și „Școala altfel”, fiecare cadru didactic are responsabilități specifice, raportate la disciplina pe care o predă, prin raportare la competențele prevăzute în programa școlară și cu reflectarea temele transversale asumate la nivel de curriculum.

Programele școlare pentru învățământul liceal:

- oferă cadrul de referință pentru proiectarea activităților specifice din aceste săptămâni, astfel încât ele să asigure legătura cu intențiile curriculumului, prin valorificarea creativă a competențelor și conținuturilor disciplinelor;
- susțin integrarea educației pentru dezvoltare durabilă, economiei circulare, educației digitale și pentru cetățenie în activități nonformale și experiențiale (proiecte, ateliere, vizite, campanii, parteneriate comunitare etc.);
- încurajează colaborarea între profesori, astfel încât activitățile din Săptămâna verde și Școala altfel să aibă coerență curriculară și impact formativ real.

În acest sens, cadrul didactic:

- proiectează și derulează activități care valorifică specificul disciplinei sale în aceste contexte, care permit abordarea aspectelor practic-aplicative ale temelor predate;
- are responsabilitatea de a asigura continuitatea formării competențelor din curriculum, în contexte nonformale;
- utilizează aceste contexte pentru a întări legătura între învățarea școlară și viața reală, între conținuturile disciplinare și temele transversale majore;
- valorifică în lecții experiența dobândită de elevi în cadrul acestor programe.

IX.9. Cadrul didactic ca actor al asigurării calității în implementarea curriculumului

Cadrul didactic este principalul actor al implementării curriculumului național și, implicit, al asigurării calității educației la nivel liceal. În acest sens, activitatea sa este sprijinită, monitorizată și analizată prin:

- mecanismele de inspecție școlară (îndrumare și monitorizare);
- evaluarea internă și externă a calității;
- programele de formare continuă;
- comunitățile profesionale de învățare.

Aplicarea programelor școlare este inseparabilă de dezvoltarea profesională continuă, de reflecția asupra propriei practici și de cooperarea în echipele didactice.

X. COERENȚA CU EVALUAREA ȘI CU STANDARDLE NAȚIONALE DE ÎNVĂȚARE

Programele școlare pentru învățământul liceal sunt concepute în deplină corelație cu sistemul de evaluare a rezultatelor învățării, astfel încât evaluarea să reflecte în mod fidel nivelul de formare a competențelor prevăzute în curriculum și să sprijine progresul elevilor.

Această coerență se realizează pe trei paliere complementare: *evaluarea formativă curentă, evaluarea sumativă de etapă și standardele naționale de evaluare.*

X.1. Programele școlare ca fundament al evaluării competențelor

Programele școlare reprezintă documentul de referință obligatoriu pentru toate tipurile de evaluare realizate la nivel liceal, deoarece:

- competențele generale și specifice definesc *ce se evaluează*;
- conținuturile indică *aria de referință a evaluării*;
- exemplele de activități de învățare și sugestiile metodologice orientează *modul de evaluare*.

Evaluarea elevilor nu se raportează la cantitatea de informații memorate, ci la *gradul de dezvoltare a competențelor*, exprimat prin:

- capacitatea de a utiliza cunoștințele în contexte variate;
- calitatea raționamentului;
- autonomia în rezolvarea sarcinilor;
- asumarea responsabilității pentru propriile demersuri.

X.2. Evaluarea ca proces continuu de reglare a învățării

Evaluarea la nivel liceal este concepută ca proces continuu, nu doar ca act de certificare, având rol de:

- diagnostic inițial al achizițiilor elevilor;
- monitorizare a progresului;
- reglare permanentă a demersului didactic;
- sprijin pentru optimizarea învățării.

În acest sens, programele școlare sunt construite astfel încât să permită:

- utilizarea unor instrumente de evaluare variate;
- corelarea permanentă dintre competențe - activități de învățare - evaluare;
- interpretarea rezultatelor evaluării în logica progresului individual al elevului.

X.3. Relația dintre programele școlare și standardele naționale de evaluare

Standardele naționale de evaluare vor fi elaborate pe baza programelor școlare liceale și vor descrie:

- nivelurile de performanță așteptate pentru fiecare competență;
- criterii și descriptori de performanță;
- praguri de atingere a competențelor la finalul unor etape de studiu.

În acest raport, programele școlare definesc *ce se învață și ce se formează*, iar standardele de evaluare definesc *cum se măsoară și cum se certifică nivelul de formare a competențelor*. Această relație asigură:

- unitatea criteriilor de evaluare la nivel național;
- echitatea în aprecierea rezultatelor elevilor;
- comparabilitatea performanțelor între unități de învățământ.

X.4. Coerența cu evaluările sumative de etapă și cu examenele naționale

Evaluările sumative de etapă și examenele naționale se fundamentează exclusiv pe competențele prevăzute în programele școlare, conținuturile asociate acestora și pe standardele naționale de evaluare. Prin urmare:

- nu pot fi evaluate achiziții care nu sunt prevăzute în programă;
- nu pot fi introduse exigențe externe curriculumului național;
- subiectele de evaluare trebuie să reflecte nivelul de competență atins de elev, nu memorarea mecanică a informațiilor.

Această coerență este esențială pentru credibilitatea sistemului de evaluare, pentru predictibilitatea evaluării pentru elevi și profesori, respectiv pentru orientarea reală a învățării către finalitățile stabilite prin curriculum.

X.5. Rolul programelor școlare în asigurarea echității evaluării

Prin formularea unitară a competențelor și a conținuturilor la nivel național, programele școlare pentru discipline obligatorii:

- asigură cadre identice de referință pentru toți elevii, indiferent de unitatea de învățământ;
- previn discrepanțele majore de exigență între școli;
- susțin principiul egalității de șanse în evaluare și certificare.

În acest fel, programele școlare contribuie direct la:

- creșterea încrederii publice în sistemul de evaluare;
- transparența criteriilor de apreciere;
- corectitudinea deciziilor educaționale luate pe baza rezultatelor evaluării.

XI. PROGRAMELE ȘCOLARE - ROLURI PENTRU ACTORII EDUCAȚIONALI

Programele școlare din învățământul liceal sunt documente curriculare normative, cu adresabilitate multiplă, fiind utilizate de categorii diferite de actori educaționali. În acest context, programele școlare pot fi valorificate diferențiat, în funcție de nevoile, responsabilitățile și deciziile fiecărei categorii de utilizatori.

Astfel, programele școlare funcționează ca *documente cu valoare sistemică*, din perspectiva „curriculumului ca bun comun”, susținând coerența întregului ecosistem educațional prin următoarele aspecte:

- oferă o interpretare coerentă și convergentă a curriculumului național, în raport cu rolurile distincte ale actorilor educaționali;
- evită confuziile între dimensiunea normativă, cea pedagogică, cea managerială și cea informativă a programelor;
- consolidează transparența, responsabilitatea și calitatea aplicării curriculumului național.

XI.1. Rolul programei școlare pentru cadrele didactice

Pentru cadrele didactice, programa școlară este *un instrument profesional de proiectare, implementare și evaluare a demersului didactic*.

În acest registru, programa este interpretată ca *document de lucru operațional*, care fundamentează:

- planificarea calendaristică anuală;
- proiectarea unităților de învățare;
- alegerea strategiilor didactice;
- construcția instrumentelor de evaluare.

XI.2. Rolul programei școlare pentru comisia de curriculum de la nivelul unității de învățământ

Pentru comisia de curriculum, programele școlare sunt documente de *referință strategică și managerială*, utilizate în scopul asigurării coerenței și calității ofertei educaționale a unității de învățământ. În acest registru, programele sunt analizate din perspectiva:

- conformității planificării și proiectării didactice cu prevederile curriculare;
- coerenței verticale și orizontale între discipline și arii curriculare;
- integrării temelor transversale la nivel instituțional;
- corelării dintre curriculum, resurse, activități extracurriculare și proiecte educaționale;
- monitorizării aplicării unitare a programelor la nivelul școlii;
- stimulării colaborării între cadrele didactice, ca principiu de bază al dezvoltării curriculare instituționale;
- susținerii proiectelor comune între discipline, între clase și între niveluri de studiu;
- promovării evaluărilor comune și a altor forme de cooperare profesională în raport cu competențele vizate.

Deciziile care privesc organizarea procesului didactic la nivelul unității de învățământ sunt fundamentate pe programele școlare astfel încât:

- să monitorizeze aplicarea unitară, completă și coerentă a curriculumului național;
- să încurajeze lucrul în echipă;
- să faciliteze abordările interdisciplinare și transdisciplinare;
- să susțină coerența parcursului educațional al elevilor;
- să deschidă școala către parteneriate, proiecte educaționale și contexte autentice de învățare.

În acest cadru, programa este utilizată ca *instrument de coordonare, articulare și dezvoltare strategică a curriculumului la nivel instituțional*, nu doar ca document de control al conformității.

XI.3. Rolul programei școlare pentru elevi și părinții lor/reprezentanții legali

Pentru elevi, programele școlare nu sunt documente tehnice, ci *repere esențiale de orientare asupra învățării, asupra propriului parcurs educațional și asupra alegerilor de traseu școlar și profesional*. Acestea oferă un cadru de înțelegere a finalităților învățării și a relevanței studiilor liceale pentru viață, continuarea studiilor și carieră.

În acest registru, programele școlare răspund elevilor și familiilor la întrebări esențiale precum:

- ce vor învăța în cadrul anumitor discipline și profiluri;
- ce competențe își vor forma pe parcursul liceului;

- care este relevanța acestor competențe pentru viața personală, pentru studiile ulterioare și pentru inserția profesională;
- care este relația dintre disciplinele studiate, profilul liceal urmat și perspectivele educaționale ulterioare.

Totodată, programele școlare se corelează cu procesul de consiliere și orientare realizat la nivelul gimnaziului, de către specialiștii din școală în colaborare cu elevii și cu părinții acestora, astfel încât:

- intențiile curriculare ale liceului să fie racordate la profilul elevului;
- deciziile privind alegerea traseului educațional să fie informate, asumate și realiste;
- tranziția de la gimnaziu la liceu să fie coerentă și susținută.

În acest sens, programa școlară sprijină dezvoltarea autonomiei elevului în învățare, asumarea responsabilității pentru propriul parcurs educațional, precum și clarificarea intereselor și a proiectului personal de viață și carieră.

XI.5. Rolul programei școlare pentru autorii de manuale și editorii de resurse educaționale

Pentru autorii de manuale școlare și pentru editorii de resurse educaționale, programele școlare liceale, împreună cu prezenta Notă de fundamentare, constituie reperul curricular obligatoriu și cadrul conceptual de proiectare a suporturilor de învățare. În acest sens, autorii și editorii se raportează prioritar la:

- competențele generale și specifice - ca finalități formative ale produsului editorial;
- domeniile de conținut - ca structură tematică de referință;
- corelarea competențe-conținuturi-activități-evaluare - ca principiu de construcție didactică;
- sugestiile metodologice - ca repere de coerență pedagogică, nu ca prescripții;
- perspectivele teoretice, metodologice și axiologice dezvoltate în prezenta Nota de fundamentare a programelor școlare - ca orizont de sens și criteriu de inovație educațională.

Manualele școlare și resursele educaționale asociate:

- nu pot introduce competențe sau conținuturi care nu sunt prevăzute în programa școlară;
- trebuie să reflecte fidel structura, logica internă și ierarhizarea curriculară prezentată în programă;
- au caracter alternativ, oferind profesorilor posibilitatea de a alege între mai multe variante pedagogice;
- trebuie să propună *elemente de noutate din perspectiva învățării*, prin:
 - strategii didactice diversificate;
 - metode activ-participative;
 - instrumente moderne de lucru;
 - integrarea rațională a tehnologiilor digitale și a inteligenței artificiale în sprijinul învățării;
- trebuie să facă *relevante și operaționale temele transversale majore*, prin activități integratoare, situații-problemă, proiecte și aplicații în contexte reale.

Astfel, manualul și resursele educaționale nu sunt concepute ca simple suporturi de transmitere a informației, ci ca *instrumente de mediere educațională între curriculum și elev*, care valorifică principiile noii paradigme curriculare: competență, transfer, integrare, reflecție, inovare și responsabilitate.

X.6. Rolul programei școlare pentru societatea civilă și partenerii educaționali

Pentru organizațiile reprezentative pentru societatea civilă, instituțiile culturale, științifice, economice și pentru alți parteneri ai școlii, programele școlare liceale reprezintă un *cadru de referință pentru inițierea, dezvoltarea și susținerea unor demersuri educaționale complementare curriculumului național*. În acest registru de lectură, programele sunt analizate din perspectiva:

- competențelor prevăzute pentru formarea elevilor;
- domeniilor de conținut care pot fi valorificate în contexte nonformale și informale;
- temelor transversale majore asumate la nivel național;
- nevoilor reale de dezvoltare ale elevilor și ale comunității.

Societatea civilă și partenerii școlii pot sprijini unitățile de învățământ în aplicarea curriculumului prin parteneriate care au ca scop:

- dezvoltarea de proiecte educaționale, culturale, civice, științifice sau tehnologice;
- organizarea de activități de învățare în contexte reale (vizite, ateliere, stagii, campanii, proiecte comunitare);
- facilitarea contactului elevilor cu mediul profesional, economic și cultural;
- susținerea integrării temelor transversale și a orientării școlare și profesionale.

În acest cadru, programele școlare funcționează ca limbaj comun și reper de coerență între școală și partenerii săi, asigurând că intervențiile externe:

- sunt compatibile cu finalitățile educaționale;
- nu fragmentează parcursul formativ al elevului, ci îl potențează;
- contribuie real la formarea competențelor prevăzute în curriculumul național.

XII. PERIOADA DE VALABILITATE A PROGRAMELOR ȘCOLARE PENTRU ÎNVĂȚĂMÂNTUL LICEAL ȘI MECANISMUL DE REVIZUIRE

Prezentele programele școlare, aprobate ca anexe la ordinul ministrului educației și cercetării, au o perioadă standard de valabilitate de 8 ani, calculată de la momentul în care prima generație de elevi parcurge integral, într-un an de studiu, programa respectivă, în condițiile aplicării noilor planuri-cadru. Această perioadă de valabilitate reflectă:

- necesitatea asigurării stabilității curriculare pe durata unui ciclu complet de liceu și pentru cel puțin încă un ciclu ulterior;
- nevoia de a permite implementarea coerentă, monitorizarea și evaluarea impactului curricular;
- exigența de a evita schimbările frecvente care pot genera discontinuități, supraîncărcare instituțională și instabilitate pedagogică.

XII.1. Ciclul de viață al programelor școlare

În logica dezvoltării curriculumului național, programele școlare parcurg un ciclu complet de viață care include următoarele etape:

1. elaborare și aprobare prin ordin de ministru;
2. implementare etapizată, în corelație cu intrarea în vigoare a noilor planuri-cadru;
3. aplicare integrală pentru prima generație de elevi;
4. monitorizare curriculară pe durata implementării;
5. evaluare sistemică a impactului la finalul perioadei de valabilitate;
6. revizuire periodică sau elaborare într-o nouă paradigmă, după caz.

Perioada de 8 ani este considerată optimă pentru maturizarea curriculară a unei programe, permițând:

- acumularea de date relevante privind eficiența formării competențelor;
- corelarea rezultatelor cu evaluările/examenelor naționale;
- formularea unor decizii fundamentate privind actualizarea conținuturilor, competențelor și abordărilor didactice.

XII.2. Monitorizarea aplicării programelor pe durata valabilității

Pe perioada de valabilitate, programele școlare fac obiectul:

- monitorizării implementării la nivel de unități de învățământ;
- analizelor realizate de structurile de specialitate din cadrul Ministerului Educației și Cercetării sau din cadrul unor instituții/unități subordonate;
- corelării cu:
 - rezultatele elevilor la evaluările de etapă și examenelor naționale;
 - rapoartele de inspecție școlară;
 - evaluările interne și externe ale calității educației;
 - analizele privind abandonul școlar, echitatea și progresul în învățare.

Monitorizarea are rolul de a identifica:

- disfuncționalități de structură curriculară;
- supraîncărcări sau redundanțe;
- dificultăți de implementare la nivel sistemic;
- eventuale necorelări sau decalaje cu evoluțiile societale, științifice sau tehnologice.

Pe parcursul celor 8 ani se pot realiza ajustări punctuale, dacă monitorizarea constată deficiențe majore de funcționare în cazul unora dintre programele școlare.

XII.3. Revizuirea programelor școlare și elaborarea de noi generații de programe

La finalul perioadei de valabilitate de 8 ani, programele școlare pot face obiectul:

- a) **revizuirii**, în situația în care:
- structura fundamentală a competențelor rămâne relevantă;
 - sunt necesare actualizări punctuale de conținut;

- se impune integrarea unor noi repere științifice, tehnologice, economice sau sociale;

b) elaborării într-o nouă paradigmă, în situația în care:

- se produc schimbări majore la nivel de:
 - legislație fundamentală a educației;
 - profil de formare al absolventului;
 - planuri-cadru;
 - paradigme educaționale;
- se constată, prin evaluare sistemică, limite structurale ale programelor existente.

Decizia privind *tipul de intervenție curriculară (revizuire sau programe noi)* se fundamentează pe:

- rapoarte de evaluare curriculară;
- analize de impact ale curriculumului național;
- consultări cu specialiști, cercetători și practicieni, după caz cu societatea civilă;
- corelarea cu noile documente de politici publice naționale și europene.

XII.4. Caracterele continuității și predictibilității curriculare

Stabilirea unei perioade de valabilitate explicite pentru programele școlare:

- garantează predictibilitatea parcursului educațional pentru elevi;
- asigură coerență instituțională pentru unitățile de învățământ;
- permite planificarea strategică a formării cadrelor didactice, a manualelor școlare și a resurselor educaționale;
- susține credibilitatea și stabilitatea curriculumului național.

Revizuirea sau înlocuirea programelor se realizează exclusiv în mod planificat, fundamentat și transparent, în cadrul unui nou ciclu de politici curriculare.

XIII. CONCLUZII - MIZA NOII GENERAȚII DE PROGRAME ȘCOLARE

XIII.1. Schimbarea de paradigmă produsă de noul pachet curricular

Noua generație de programe școlare pentru învățământul liceal continuă reforma curriculară implementată deja la nivelul educației timpurii și al învățământului primar și gimnazial, marcând trecerea:

- de la un curriculum centrat predominant pe conținuturi la un curriculum centrat pe competențe, transfer și sensul învățării;
- de la o viziune fragmentată, disciplinară, la una integrată, coerentă și orientată spre profilul absolventului;
- de la accentul pe predare, la accentul pe învățare activă, reflexivă și aplicativă;
- de la o abordare uniformizatoare, la una care susține diferențierea și personalizarea parcursurilor educaționale;
- de la separarea rigidă între educația formală și realitățile societale, la o deschidere explicită către viața reală, către problemele contemporane și către viitor.

Programele școlare devin astfel nu doar documente de reglementare, ci instrumente de politică educațională activă, cu rol de orientare a practicilor didactice, a evaluării și a culturii școlare.

XIII.2. Impactul asupra formării elevului

Din perspectiva elevilor din învățământul liceal, implementarea noilor programe școlare are ca efect:

- formarea unor competențe durabile, transferabile și funcționale, nu doar acumularea de cunoștințe;
- dezvoltarea gândirii critice, a capacității de analiză, de decizie și de rezolvare de probleme;
- consolidarea autonomiei în învățare și a responsabilității personale;
- formarea unei conștiințe civice, etice și ecologice;
- sprijinirea orientării școlare și profesionale informate;
- creșterea relevanței școlii pentru viața reală a elevilor și pentru traseele lor viitoare educaționale și profesionale.

Astfel elevul nu mai este privit doar ca beneficiar al unui volum de informații, ci ca **subiect activ al propriei formări**, capabil să înțeleagă, să aplice, să evalueze și să transfere ceea ce învață.

XIII.3. Impactul asupra profesiei didactice

Profesia didactică, în acord și cu profilul de formare și competențele profesionale ale cadrelor didactice, este repositionată ca *profesie reflexivă, responsabilă și strategică pentru dezvoltarea societății*, nu doar ca rol de execuție curriculară. Astfel, noua generație de programe școlare liceale:

- consolidează statutul profesional al cadrului didactic ca proiectant al învățării, nu doar ca transmițător de conținuturi;
- valorizează autonomia profesională, exercitată în cadrul unor repere curriculare clare și asumate;
- stimulează colaborarea între profesori, proiectele comune și evaluările corelate;
- susține dezvoltarea unei culturi a reflecției asupra practicii didactice;
- impune necesitatea formării continue, în special în domeniul:
 - evaluării competențelor;
 - utilizării tehnologiilor și a inteligenței artificiale;
 - educației pentru dezvoltare durabilă și cetățenie.

XIII.4. Impactul asupra relației școală-societate

Prin orientarea explicită spre competențe, teme transversale și realități contemporane, școala este astfel reafirmată ca instituție fundamentală de construcție socială, nu doar ca spațiu de instruire, prin faptul că noile programe școlare:

- apropie școala de viața socială, economică și civică;
- facilitează parteneriatele dintre unitățile de învățământ și comunitate;
- consolidează încrederea publică în relevanța educației;
- sprijină formarea unui cetățean activ, informat, responsabil și adaptabil;
- contribuie la dezvoltarea unei societăți bazate pe:
 - cunoaștere;
 - responsabilitate;

- solidaritate;
- sustenabilitate;
- respect pentru valori democratice.

XIII.5. Noile programe școlare pentru învățământul liceal - pilon esențial al reformei curriculare naționale

Noile programe școlare pentru învățământul liceal reprezintă un pilon esențial al reformei curriculare naționale, prin care se creează condițiile pentru:

- formarea unor absolvenți competenți, responsabili și adaptabili;
- consolidarea unei profesii didactice moderne și reflexive;
- întărirea legăturii dintre educație și societate.

Prin coerența sa internă, prin raportarea clară la profilul de formare al absolventului și prin deschiderea către provocările secolului XXI, acest pachet curricular constituie o investiție strategică în viitorul educației românești.

Ministerul Educației și Cercetării adresează mulțumiri și recunoștință tuturor membrilor grupurilor de lucru care au contribuit la elaborarea acestei noi generații de programe școlare liceale. Prin expertiza profesională, rigoarea științifică, responsabilitatea publică și efortul susținut depus pe parcursul acestui amplu proces, aceștia au făcut posibilă construcția unui cadru curricular coerent, modern și relevant pentru formarea tinerilor. Contribuția lor reflectă un angajament autentic față de calitatea educației și față de viitorul școlii românești, iar acest efort colectiv reprezintă o expresie a solidarității profesionale și a dorinței comune de a servi interesul public.

| Număr anexei la ordin | Disciplină | Tip programă |
|---|----------------------------|--|
| | | |
| 41. | EDUCAȚIE VIZUALĂ | Programa școlară pentru disciplina <i>Educație vizuală</i> ; <ul style="list-style-type: none"> - Clasele a IX-a și a X-a, trunchi comun (TC)/ - Clasele a IX-a și a X-a, curriculum de specialitate (CS), filiera vocațională, profil artistic, specializarea arta actorului - Clasa a XI-a (TC) și clasa a XII-a (CDEOȘ), filiera vocațională, profil artistic, specializarea muzică |
| C. PROGRAME ȘCOLARE PENTRU DISCIPLINE OBLIGATORII CU ALOCARE ORARĂ ÎN CURRICULUM DE SPECIALITATE PENTRU: FILIERA TEORETICĂ, TOATE PROFILURILE ȘI SPECIALIZĂRILE; FILIERA TEHNOLOGICĂ, TOATE PROFILURILE ȘI CALIFICĂRILE PROFESIONALE; FILIERA VOCAȚIONALĂ, PROFILURILE MILITAR, PEDAGOGIC ȘI SPORTIV, TOATE SPECIALIZĂRILE | | |
| 42. | INFORMATICĂ | Programa școlară pentru disciplina <i>Informatică</i> , clasa a IX-a, curriculum de specialitate (CS), filiera teoretică, profilul real, specializarea matematică-informatică |
| 43. | INFORMATICĂ | Programa școlară pentru disciplina <i>Informatică</i> , clasa a IX-a, curriculum de specialitate (CS), filiera teoretică, profilul real, specializarea științe ale naturii |
| 44. | LIMBA LATINĂ | Programa școlară pentru disciplina <i>Limba latină</i> , clasele a IX-a – a XII-a, curriculum de specialitate (CS), filiera teoretică, profilul umanist, specializarea filologie și specializarea științe sociale |
| 45. | CURRICULUM DE SPECIALITATE | Curriculum pentru clasa a IX-a, învățământ liceal, filiera tehnologică, domeniul de pregătire profesională <i>agricultură</i> |
| 46. | CURRICULUM DE SPECIALITATE | Curriculum pentru clasa a IX-a, învățământ liceal, filiera tehnologică, domeniul de pregătire profesională <i>chimie industrială</i> |
| 47. | CURRICULUM DE SPECIALITATE | Curriculum pentru clasa a IX-a, învățământ liceal, filiera tehnologică, domeniul de pregătire profesională <i>comerț</i> |
| 48. | CURRICULUM DE SPECIALITATE | Curriculum pentru clasa a IX-a, învățământ liceal, filiera tehnologică, domeniul de pregătire profesională <i>construcții, instalații și lucrări publice</i> |
| 49. | CURRICULUM DE SPECIALITATE | Curriculum pentru clasa a IX-a, învățământ liceal, filiera tehnologică, domeniul de pregătire profesională <i>economic</i> |
| 50. | CURRICULUM DE SPECIALITATE | Curriculum pentru clasa a IX-a, învățământ liceal, filiera tehnologică, domeniul de pregătire profesională <i>electric</i> |
| 51. | CURRICULUM DE SPECIALITATE | Curriculum pentru clasa a IX-a, învățământ liceal, filiera tehnologică, domeniul de pregătire profesională <i>electromecanică</i> |

Programa școlară
pentru disciplina

Informatică

Clasa a IX-a

Curriculum de specialitate (CS)

Filiera teoretică, profilul real, specializarea matematică-informatică

Învățământ liceal

Anexa nr. 42**Programa școlară pentru disciplina Informatică, clasa a IX-a, curriculum de specialitate (CS)**

- *Filiera teoretică, profilul real, specializarea matematică-informatică*

NOTĂ DE PREZENTARE**Statutul disciplinei**

Disciplina *informatică*, studiată în ciclul liceal, este o continuare a disciplinei *informatică și TIC*, studiată la gimnaziu, în trunchiul comun. În ciclul liceal, aceasta își consolidează și extinde domeniile de competență, aprofundând aspectele conceptuale, algoritmice și aplicative ale domeniului informatic.

Conform Ordinului ministrului educației și cercetării nr. 4.350/2025 privind aprobarea planurilor-cadru pentru învățământul liceal cu frecvență zi, disciplina *informatică* se predă ca disciplină din categoria curriculumului de specialitate (CS) la:

- filiera teoretică, profilul real, specializarea matematică - informatică, în clasele a IX-a, a X-a, a XI-a și a XII-a;
- filiera teoretică, profilul real, specializarea științe ale naturii, în clasele a IX-a și a X-a;
- filiera vocațională, profilul militar, specializarea matematică - informatică militară, în clasele a IX-a, a X-a, a XI-a și a XII-a.

Pentru filiera teoretică, profilul real, specializarea matematică - informatică, alocarea orară este:

- clasa a IX-a – 2 ore/săptămână, dintre care o oră pentru studiu teoretic și o oră pentru activități practice;
- clasa a X-a – 2 ore/săptămână, dintre care o oră pentru studiu teoretic și o oră pentru activități practice;
- clasa a XI-a – 4 ore/săptămână, dintre care două ore pentru studiu teoretic și două ore pentru activități practice;
- clasa a XII-a – 3 ore/săptămână, dintre care o oră pentru studiu teoretic și două ore pentru activități practice.

Activitățile practice sunt desfășurate obligatoriu în laboratorul de informatică.

Conform planurilor cadru, în învățământul liceal, componenta fixă a curriculumului de specialitate **poate fi completată cu o componentă flexibilă**, ce include discipline/module de specialitate și alocări orare la decizia unității de învățământ; curriculumul de specialitate poate fi sprijinit/completat cu opționale de aprofundare, noi discipline sau poate fi valorizat din perspectivă integrată/interdisciplinară, prin oferta națională de opționale sau prin oferta unității de învățământ.

La acesta se adaugă **curriculumul la decizia elevului din oferta școlii** (CDEOS), prin opționale care pot fi de aprofundare, noi discipline/module de pregătire sau opționale integrate prin care elevii pot primi sprijin pentru remediare, consolidare sau performanță, pentru pregătirea suplimentară, respectiv pentru pregătirea susținerii atestatelor de certificare a competențelor profesionale sau pentru satisfacerea unor nevoi și interese pentru specializare la nivel superior, inclusiv pentru admiterea la studii universitare.

Raportarea la cadrul legislativ și documentele strategice generale și specifice care susțin/întemeiază studiul disciplinei

Elaborarea prezentei programe școlare este bazată pe documente fundamentale care definesc viziunea și structura curriculumului național:

- Legea învățământului preuniversitar nr. 198/2023, cu modificările și completările ulterioare, precum și alte acte subsecvente relevante privind implementarea curriculumului național;
- Ordinul privind aprobarea planurilor-cadru pentru învățământul liceal cu frecvență zi (Ordinului ministrului educației și cercetării nr. 4.350/2025);
- Recomandarea Consiliului UE privind competențele-cheie pentru învățarea pe tot parcursul vieții (2018);
- Cadrul european al calificărilor (EQF);
- Rapoarte OECD/UNESCO privind competențele digitale și educația STEM, Cadre de referință;
- Profilul de formare al absolventului (Ordinul ministrului educației 6731/2023);
- Cadrul european al competențelor digitale pentru cetățeni (DigComp), respectiv Cadrul de Competențe digitale pentru elevi DigiComp 2.2. (Anexa_OM_6466_2024).

Rolul disciplinei în formarea elevilor

Studiul disciplinei *informatică* vizează formarea unei gândiri computaționale, algoritmice, analitice și creative, capabile să abordeze probleme complexe prin modele și instrumente specifice științei calculatoarelor, valorificând competențele formate pe parcursul ciclului gimnazial. Informatica se bazează pe o gândire riguroasă, o capacitate de abstracție și modelare, dar și pe utilizarea tehnologiilor digitale în contexte variate — academice, profesionale și cotidiene.

Disciplina contribuie direct la realizarea profilului de formare al absolventului, dezvoltând competențe-cheie definite la nivel european, cum ar fi în principal, competența digitală, competența matematică și competența în științe, tehnologie și inginerie, dar, indirect, și celelalte competențe cheie europene, prin activități de învățare adecvate și utilizarea limbajului de specialitate în diferite contexte.

„Ideile mari” promovate de disciplină vizează dezvoltarea gândirii computaționale, rezolvarea problemelor de natură informatică, elaborarea unor algoritmi eficienți, scrierea codului clar și funcțional, analiza complexității soluțiilor propuse, precum și interacțiunea cu diferite modele conceptuale de organizare a datelor.

Utilitatea studiului disciplinei *informatică* se manifestă pe multiple planuri:

- pe parcursul școlar, prin fundamentarea gândirii computaționale, necesare și pentru alte discipline;
- în viața cotidiană, prin planificarea riguroasă a acțiunilor, utilizarea critică și creativă a instrumentelor digitale;
- în formarea profesională, prin deschiderea către cariere în IT, automatizare, robotică, analiză de date, securitate cibernetică sau inteligență artificială.

Justificarea statutului disciplinei *informatică*, elemente de continuitate/de noutate

Disciplina *informatică* valorifică achizițiile formate în gimnaziu la nivelul competențelor digitale de bază și al utilizării instrumentelor informatice, aducând progres prin abordarea formală și științifică a modelelor de organizare a datelor, a algoritmilor specializați pe tipuri de probleme, programarea structurată și orientată pe obiecte, eficiența rezolvărilor, precum și elemente de inteligență artificială și baze de date.

Caracterul său de curriculum de specialitate (CS) subliniază rolul central al disciplinei în formarea competențelor profesionale ale elevilor în domeniul informatic. Prin natura sa, informatica se află la intersecția între cunoaștere teoretică și aplicare practică, consolidând gândirea logică și deschiderea către știință, inovație și responsabilitate digitală.

Categorii de programe școlare pentru disciplina *informatică*

Disciplina *informatică* se încadrează în categoria Curriculumului de specialitate (CS), fiind destinată aprofundării competențelor specifice.

Orientări generale și specifice în lectura programei școlare

Aplicarea programei presupune:

- respectarea competențelor generale și specifice ca repere obligatorii în proiectarea activităților didactice;
- proiectarea didactică flexibilă, centrată pe situații de învățare autentice și evaluări bazate pe competențe;
- corelarea competențelor generale și specifice cu domeniile de conținut și cu exemplele de activități de învățare;
- planificarea integrată a conținuturilor, cu accent pe rezolvarea de probleme, lucrul în echipă, proiecte interdisciplinare și utilizarea resurselor digitale moderne;
- valorificarea componentelor orientative pentru adaptarea la particularitățile colectivului de elevi și la resursele școlii;
- utilizarea Python ca limbaj de programare de bază și a limbajelor C++, respectiv a SQL, conform programei școlare corespunzătoare profilului, clasei și specializării;
- utilizarea unor resurse digitale moderne în laboratoare de informatică dotate adecvat;
- corelarea activităților de învățare cu domenii STEM, economie, științe sociale și arte digitale.

Programa are caracter obligatoriu în ceea ce privește competențele generale, competențele specifice și conținuturile precizate, iar componentele metodologice și exemplele de activități de învățare au caracter orientativ, oferind cadrul pentru adaptarea la nivelul clasei și al resurselor disponibile. Profesorul are libertatea de a selecta și combina metode, resurse și instrumente digitale adecvate, cu condiția respectării finalităților și competențelor prevăzute de programă. Se recomandă accentuarea caracterului practic, formativ și explorator al disciplinei, pentru a consolida autonomia elevului în învățare și în formarea unei culturi informatice autentice.

Alegerea limbajului Python ca limbaj de programare de bază în studiul disciplinei *informatică* răspunde cerințelor unui învățământ modern, centrat pe formarea gândirii algoritmice și a competențelor digitale relevante. Datorită sintaxei sale simple și clare, Python facilitează înțelegerea conceptelor fundamentale de programare, permițând elevilor să se concentreze pe rezolvarea problemelor și pe dezvoltarea logicii algoritmice. Totodată, prin caracterul său actual și aplicativ, Python este utilizat pe scară largă în domenii precum analiza datelor, inteligența artificială, automatizare și dezvoltare software, oferind elevilor o pregătire relevantă pentru continuarea studiilor și integrarea profesională.

Studiul limbajului C++ are o valoare formativă și permite aprofundarea conceptelor fundamentale de programare și a gândirii algoritmice. După însușirea principiilor de bază în Python, C++ oferă elevilor oportunitatea de a înțelege mai profund mecanismele interne ale programării, precum gestionarea memoriei și structurile de date dinamice.

Introducerea noțiunilor de învățare automată (Machine Learning) în programa școlară corespunzătoare clasei, profilului și specializării, este importantă pentru familiarizarea elevilor cu una dintre cele mai importante ramuri ale informaticii moderne. Studiul învățării automate permite elevilor să înțeleagă cum pot fi antrenate modelele și algoritmii pentru a recunoaște tipare și a realiza predicții pe baza datelor, folosind limbajul Python și biblioteci specifice. Prin activități practice, precum clasificarea imaginilor, analiza datelor sau predicția unor valori, elevii dobândesc competențe reale de analiză și programare aplicată. Învățarea automată contribuie astfel la dezvoltarea gândirii logice și algoritmice, oferind o bază solidă pentru studii superioare și cariere în domenii precum informatică, știința datelor sau inteligența artificială.

Sistemele informatice moderne (site-uri web, aplicații, platforme educaționale) depind de baze de date pentru a funcționa automatizat. Studiul bazelor de date este important în formarea competențelor digitale, deoarece acestea reprezintă fundamentul gestionării eficiente a datelor în orice domeniu de activitate. Într-o societate bazată pe date, capacitatea de a colecta, organiza, analiza și interpreta informații este esențială pentru viața profesională și personală.

Setul de competențe generale ale disciplinei *informatică* este construit pe baza taxonomiei Bloom (revizuite), urmărind o dezvoltare progresivă a gândirii logice și algoritmice, de la nivelurile de înțelegere și aplicare până la cele de analiză, evaluare și

creare. Această structură sprijină elevii în formarea unei viziuni integrate asupra procesului de dezvoltare software, de la concept la implementare, asigurând totodată experiență de învățare în domeniul informaticii.

Prin această abordare, competențele generale facilitează o evoluție cognitivă clară: de la identificarea și explicarea modelelor conceptuale și operaționale care stau la baza programării, la aplicarea și analiza acestora în contexte variate, continuând cu evaluarea critică a soluțiilor informatice și crearea de produse software originale, adaptate cerințelor.

În ansamblu, competențele generale precizate în programă contribuie la formarea competențelor digitale, logice și creative necesare, ca bază, unui specialist în domeniul informatic, într-o societate bazată pe tehnologie și inovare.

Programa școlară de *informatică* este calibrată astfel încât să asigure o rezervă de 25% din timpul alocat disciplinei, la dispoziția cadrului didactic pentru activități de remediere, consolidare, aprofundare sau extindere.

Structura programei școlare include, pe lângă Nota de prezentare, următoarele componente:

- Competențe generale;
- Competențe specifice și exemple de activități de învățare;
- Conținuturi;
- Sugestii metodologice.

Competențele generale (CG) vizate la nivelul disciplinei integrează achizițiile de cunoaștere și de comportament așteptate, subliniind orientarea generală a procesului educațional la această disciplină.

Competențele generale sunt derivate din competențele-cheie și explicitează finalitățile majore ale disciplinei, acele achiziții de durată pe care toți elevii trebuie să le dobândească prin întreg studiul acesteia, la nivelul ciclului liceal. Acestea dau coerență disciplinei, stabilesc direcția învățării și fundamentează derivarea competențelor specifice, selecția și organizarea conținuturilor învățării. Competențele generale au grad ridicat de complexitate și integrează ansambluri de cunoștințe, abilități și atitudini, ca rezultate ale învățării utile pentru dezvoltarea personală, pentru cetățenia activă, pentru incluziune socială și pentru angajare pe piața muncii.

Competențele specifice (CS) sunt competențe derivate din competențele generale și reprezintă etape măsurabile în formarea și dezvoltarea acestora, ilustrând rezultate ale învățării pentru fiecare an de studiu. Acestea exprimă, pentru elevi, achizițiile învățării prin parcurgerea disciplinei de studiu, și includ, la fel ca în cazul competențelor generale, ansambluri de cunoștințe, abilități și atitudini. Competențele specifice asigură continuitatea de la gimnaziu, progresia de la un an la altul și conexiunea cu profilul de formare al absolventului.

Pentru formarea și dezvoltarea competențelor specifice, în programă sunt propuse **exemple de activități de învățare (EAI)**, care descriu contexte și modalități prin care competențele specifice sunt formate, exersate, consolidate și evaluate în mod curent, formativ. Ele au rol orientativ, nu prescriptiv, și oferă profesorilor repere privind modul în care pot organiza situații de învățare relevante pentru elevi. Astfel, profesorul poate să adapteze activitățile de învățare propuse în programă, să le completeze sau să le înlocuiască cu altele adecvate clasei, asigurând cadrul unui demers didactic personalizat, pentru formarea/dezvoltarea competențelor prevăzute de programă, în contextul specific al fiecărei clase.

Conținuturile sunt organizate în domenii de conținut (categorii mari) și, în cadrul acestora, în conținuturi propriu-zise ale învățării. Domeniile de conținut și conținuturile învățării definesc „substanța” disciplinei: ce anume se studiază efectiv, pentru a sprijini formarea competențelor. Acestea constituie o selecție, adecvată din punctul de vedere didactic, de elemente din domeniul de studiu al disciplinei (informații factuale, conceptuale, procedurale), cu rol de suport operațional/instrumental pentru formarea competențelor specifice. Selecția este făcută pe baza principiului continuității și al coerenței, iar conținuturile sunt interconectate, astfel încât, după parcurgerea lor integrală, elevul să fie capabil să realizeze conexiuni, în scopul rezolvării unor probleme diverse, de natură teoretică sau practic-aplicativă.

Sugestiile metodologice au rolul de a sprijini profesorii în aplicarea programei, fără a impune sau a face o inventariere a metodelor didactice utilizate. Acestea traduc intențiile programei (CG, CS, conținuturi, exemple de activități de învățare) în modalități și mijloace pentru realizarea demersului didactic, prin exemple minimale, relevante, de abordare a activității didactice, pentru alegerea strategiilor didactice și pentru integrarea conținuturilor și competențelor în practica școlară.

Astfel, programa școlară oferă un cadru coerent de utilizare: competențele generale indică direcțiile de învățare, competențele specifice, pe ani de studiu, precizează etapele de progres, domeniile de conținut stabilesc suportul științific pentru formarea acestor competențe, iar exemplele de activități de învățare ilustrează modalități concrete de dezvoltare a experiențelor de învățare.

COMPETENȚE GENERALE (CG)

| | |
|------------|---|
| CG1 | Identifică principalele caracteristici ale modelelor conceptuale și operaționale ale dezvoltării produselor software, pentru înțelegerea fundamentelor programării |
| CG2 | Explică principii care stau la baza modelelor conceptuale și operaționale ale dezvoltării produselor software, pentru a fundamenta în mod logic proiectarea și implementarea soluțiilor informatice |
| CG3 | Utilizează modele conceptuale și operaționale ale dezvoltării produselor software, în scopul obținerii de soluții informatice funcționale și eficiente |
| CG4 | Analizează caracteristicile și aplicabilitatea modelelor conceptuale și operaționale ale dezvoltării produselor software, pentru a selecta soluțiile cele mai potrivite în funcție de contextul informatic dat |
| CG5 | Evaluează corectitudinea și eficiența soluțiilor informatice, în vederea optimizării și asigurării funcționalității în diverse scenarii de utilizare |
| CG6 | Elaborează algoritmi și programe personalizate, pentru a crea soluții informatice coerente și adaptate cerințelor |

COMPETENȚE SPECIFICE (CS) ȘI EXEMPLE DE ACTIVITĂȚI DE ÎNVĂȚARE (EAI)

CG 1 - Identifică principalele caracteristici ale modelelor conceptuale și operaționale ale dezvoltării produselor software, pentru înțelegerea fundamentelor programării

| Clasa a IX-a | |
|---|--|
| CS 1.1. Identifică principalele caracteristici ale organizării datelor în cadrul unor modele conceptuale fundamentale - date simple sau liste, pentru a structura și accesa date în vederea prelucrării acestora | |
| <ul style="list-style-type: none"> - recunoașterea caracteristicilor unei liste evidențiind modul în care se succed mașinile care sunt oprite la semafor pe o stradă cu sens unic, faptul că fiecare mașină, cu excepția primei și ultimei mașini, are imediat înaintea ei și imediat după ea câte o altă mașină - enumerarea reperelor pentru parcurgerea unei liste în vederea identificării perechilor de elemente aflate pe poziții consecutive și care au aceleași caracteristici - recunoașterea caracteristicilor unei liste cu acces direct prin nominalizarea celui de al cincilea elev din catalog, fără a fi necesară parcurgerea tuturor datelor elevilor care îl preced în catalog - recunoașterea caracteristicilor unei stive și ale unei cozi, din punctul de vedere al modului în care se accesează și gestionează elementele, în contextul exemplificării stivuirii unor farfurii, respectiv așezării unor persoane la un rând, pentru a cumpăra bilete la cinema | |
| CS 1.2. Identifică specificul, caracteristicile și etapele unor algoritmi specializați pe clase de probleme, pentru a îi putea utiliza în prelucrarea algoritmică a numerelor, sortarea sau generarea sistematică a unor secvențe de valori | |
| <ul style="list-style-type: none"> - enumerarea etapelor algoritmului lui Euclid pentru determinarea celui mai mare divizor comun cu scopul determinării dimensiunii maxime a unei plăci de gresie de formă pătrată folosită pentru acoperirea completă, cu plăci întregi, a podelei unei săli de dimensiuni date - enumerarea etapelor algoritmului de transformare în baza 2 a unui număr scris în baza 10 în contextul reprezentării în memorie a unui număr natural (de exemplu, codul ASCII asociat unei litere) - reamintirea secvenței de operații necesare pentru a adăuga o cifră la dreapta celorlalte cifre ale unui număr, în contextul determinării valorii rezultate în urma eliminării tuturor cifrelor impare ale unui număr, evidențiind necesitatea păstrării ordinii inițiale a cifrelor pare rămase și a cifrelor nule de la finalul numărului - reamintirea regulii de obținere a unui termen al șirului Fibonacci, pe baza celor doi termeni anteriori, în contextul unui proiect de cercetare având ca temă șirul lui Fibonacci în natură - identificarea caracteristicilor metodei de sortare prin selecția minimului aplicată scenariului de aranjare a finaliștilor unui concurs de talente în ordinea crescătoare a scorului obținut - recunoașterea caracteristicilor metodei bulelor (compararea elementelor adiacente, interschimbarea celor aflate în ordine necorespunzătoare) aplicată unui context real, precum ordonarea înălțimilor elevilor unei clase | |
| CS 1.3. Identifică specificul, caracteristicile și etapele unor strategii de rezolvare a problemelor prin proiectarea modulară a algoritmilor | |
| <ul style="list-style-type: none"> - identificarea datelor cu care lucrează algoritmii, pe baza caracteristicilor acestora (date de intrare, date ieșire, date de manevră) - conceperea unei diagrame care ilustrează etapele de elaborare a unui program - enumerarea etapelor de elaborare a unui program, precizând rolul fiecăreia - asocierea unor termeni precum modul, gândire computațională, proiectare cu rolul lor în rezolvarea unei probleme informatice | |
| CS 1.4. Identifică principalele elemente ale limbajului de programare utilizate pentru prelucrarea datelor organizate în modele fundamentale - date simple sau liste, respectiv pentru transmiterea datelor de la și către un program, în vederea rezolvării eficiente a problemelor informatice | |
| <ul style="list-style-type: none"> - recunoașterea operatorilor specifici clasei list din Python ([], in, not in, +, *, ==, !=, <, >), în contextul recapitulării modalităților de lucru cu structuri de date, prin completarea unui tabel care asociază fiecare operator cu rolul său - descrierea unor metode suplimentare ale clasei list din Python (extend(), reverse(), clear(), remove(), sort(), copy()), în contextul recapitulării instrumentelor de prelucrare a colecțiilor de date, prin completarea unui tabel care asociază fiecare metodă cu acțiunea sa principală asupra listei - enumerarea etapelor în lucrul cu un fișier text și a metodelor corespunzătoare în Python: open(), read(), write() și close() - identificarea pe un exemplu de program a subprogramelor asociate butoanelor sau evenimentelor de tip "apăsarea unei taste" | |

| Clasa a IX-a |
|--|
| CS 1.5. Identifică elementele de sintaxă și componentele din definiția și apelul subprogramelor și subprogramelor predefinite pentru operații uzuale, pentru a le utiliza în implementarea algoritmilor în limbaj de programare |
| <ul style="list-style-type: none"> - descrierea structurii de bază a unui subprogram în Python (definire cu <code>def</code>, parametri, instrucțiuni, apel) în contextul recapitulării modului de organizare a codului într-un proiect existent, prin completarea unui șablon de cod parțial - recunoașterea funcțiilor predefinite Python pentru calcule matematice uzuale (<code>abs()</code>, <code>round()</code>, <code>int()</code>, <code>sqrt()</code>), într-un program dat, de rezolvare a unei probleme de calcul numeric - identificarea funcțiilor predefinite Python utilizate pentru colecții (<code>len()</code>, <code>min()</code>, <code>max()</code>, <code>sum()</code>), în contextul recapitulării elementelor de bază privind manipularea listelor numerice, prin completarea unui tabel care asociază fiecare funcție cu rezultatul obținut pentru o listă dată |

CG 2 - Explică principii care stau la baza modelelor conceptuale și operaționale ale dezvoltării produselor software, pentru a fundamenta în mod logic proiectarea și implementarea soluțiilor informatice

| Clasa a IX-a |
|--|
| CS 2.1. Explică principiile de organizare a datelor în cadrul unor modele conceptuale fundamentale - date simple sau liste, pentru a le gestiona eficient |
| <ul style="list-style-type: none"> - explicarea modului în care principiile de organizare ale unei stive și ale unei cozi reflectă situații din viața reală, exemplificate prin stiva de farfurii dintr-o bucătărie (LIFO) și coada de persoane la ghișeu (FIFO), pentru a evidenția diferențele de acces - explicarea rolului unei liste de frecvențe în organizarea și interpretarea datelor dintr-o situație reală, exemplificată printr-o listă care centralizează voturile obținute de mai mulți candidați, interpretând semnificația valorilor înregistrate în listă ca frecvențe ale opțiunilor exprimate - explicarea modului în care parcurgerea secvențială a datelor permite extragerea de informații esențiale dintr-o colecție organizată liniar, exemplificând prin numărarea persoanelor care au participat la un eveniment, pe baza unui criteriu dat - explicarea necesității memorării datelor într-o listă pentru a permite prelucrări ulterioare, exemplificând prin înregistrarea zilnică a numărului de clienți ai unui magazin într-o lună pentru a identifica zilele cu numărul minim de clienți |
| CS 2.2. Explică etapele algoritmilor specializați pe clase de probleme, pentru a îi putea utiliza în prelucrarea algoritmică a numerelor, sortarea sau generarea sistematică a unor secvențe de valori |
| <ul style="list-style-type: none"> - explicarea procesului de formare a unui număr nou prin adăugarea repetată a câte unei cifre în stânga celor existente, evidențiind înmulțirea cifrei adăugate cu o variabilă, pe baza unui algoritm dat - explicarea modului de formare a unei liste de valori ai cărei termeni respectă o anumită regulă (de exemplu, lista temperaturilor medii anuale în procesul de încălzire globală știind temperaturile din ani consecutivi) - explicarea aplicabilității proporției de aur în știință, artă și în natură, pe baza materialelor de documentare primite, privind șirul lui Fibonacci - explicarea procesului de aranjare în ordine crescătoare a șapte comenzi online în funcție de valoarea comenzilor, folosind selecția minimului, cu reprezentarea ordinii comenzilor după fiecare parcurgere |
| CS 2.3. Explică etapele unor strategii de rezolvare a problemelor prin proiectarea modulară a algoritmilor |
| <ul style="list-style-type: none"> - explicarea etapelor a două strategii de rezolvare prin proiectarea modulară a algoritmilor pentru aceeași problemă, subliniind diferențele de organizare a modulelor și impactul acestora asupra clarității și eficienței soluției - explicarea rolului și importanței fiecărei etape din procesul de dezvoltare a unui program - explicarea rolului fiecărui modul în cadrul unei aplicații cunoscute, pentru realizarea obiectivelor acesteia |
| CS 2.4. Explică rolul principalelor elemente ale limbajului de programare utilizate pentru prelucrarea datelor organizate în modele fundamentale - date simple sau liste, respectiv pentru transmiterea datelor de la și către un program în vederea rezolvării eficiente a problemelor informatice |
| <ul style="list-style-type: none"> - explicarea modului în care caracteristicile clasei list (mutabilitate, acces prin index, ordinea inserării) influențează manipularea datelor, în contextul unei aplicații de gestionare a notelor elevilor, prin descrierea pașilor de adăugare, modificare și ștergere a elementelor - explicarea modului de funcționare a metodelor <code>append()</code>, <code>insert()</code> și <code>pop()</code> în contextul unei aplicații de gestionare a unei liste de participanți la un eveniment, prin justificarea efectelor fiecărei metode asupra structurii listei - explicarea mecanismelor de citire și scriere într-un fișier și a noțiunii de "sfârșit de fișier" - explicarea rolului principalelor obiecte grafice într-o aplicație mai complexă, dată, care utilizează cât mai multe obiecte grafice (fereastră, etichetă, buton, casetă de text, listbox, messagebox, meniu) |
| CS 2.5. Explică mecanismul de executare a subprogramelor și subprogramelor predefinite pentru operații uzuale, pentru a le utiliza în implementarea algoritmilor în limbaj de programare |
| <ul style="list-style-type: none"> - explicarea rolului subprogramelor într-un proiect de organizare modulară a unui algoritm dat, care determină câte numere dintr-un șir de valori au suma cifrelor un număr prim, prin descrierea relației dintre antet, parametri și corpul fiecărui subprogram - explicarea modului de funcționare a apelului unui subprogram în contextul unei aplicații de calcul al costului total al cumpărăturilor, prin descrierea legăturii dintre parametri, valoarea returnată și afișarea rezultatului |

| Clasa a IX-a |
|---|
| <ul style="list-style-type: none"> - explicarea modului de funcționare a subprogramelelor predefinite pentru conversii și rotunjiri, în contextul unui exemplu practic de prelucrare a notelor școlare, prin precizarea diferenței dintre <code>int()</code> și <code>round()</code> - explicarea modului de funcționare a programelor ce utilizează funcțiile <code>len()</code>, <code>min()</code>, <code>max()</code> și <code>sum()</code> în contextul unei aplicații de gestiune a notelor elevilor, prin argumentarea legăturii dintre fiecare funcție și informația pe care o furnizează despre colecție |

CG 3 - Utilizează modele conceptuale și operaționale ale dezvoltării produselor software, în scopul obținerii de soluții informatice funcționale și eficiente

| Clasa a IX-a |
|--|
| CS 3.1. Utilizează modul de organizare și prelucrare a datelor în cadrul unor modele conceptuale fundamentale - date simple sau liste, pentru a rezolva probleme de gestionare a datelor |
| <ul style="list-style-type: none"> - aplicarea modului de funcționare al stivei și al cozii în organizarea sarcinilor zilnice, prin simularea unei liste de activități unde ultima activitate adăugată este prima activitate rezolvată (stivă) sau prima activitate adăugată este prima activitate executată (coadă), evidențiind relevanța alegerii tipului de acces în funcție de context - aplicarea conceptului de listă de frecvențe în analiza răspunsurilor dintr-un chestionar de satisfacție, prin numărarea aparițiilor fiecărei variante de răspuns (1–5 stele) și organizarea rezultatelor sub formă de listă pentru identificarea tendințelor generale - aplicarea principiilor parcurgerii liniare pentru folosirea valorilor care respectă o anumită condiție într-un context real, cum ar fi verificarea listelor de plăți lunare pentru determinarea facturilor restante, prin procesarea secvențială a fiecărui element fără memorarea rezultatelor intermediare - aplicarea conceptului de parcurgere liniară cu memorare prin stocarea notelor obținute de elevii dintr-o clasă la un test, utilizând lista pentru determinarea elevilor care au notele mai mari decât media clasei |
| CS 3.2. Aplică algoritmi specializați pe clase de probleme, pentru a îi putea utiliza în prelucrarea algoritmică a numerelor, sortarea sau generarea sistematică a unor secvențe de valori |
| <ul style="list-style-type: none"> - simularea procesului de obținere a divizorilor unui număr natural dat utilizând reperele pentru parcurgerea acestora - implementarea unor algoritmi de divizibilitate dați pentru a verifica dacă un număr natural este perfect, determinând suma divizorilor acestuia - reprezentarea în pseudocod a algoritmului lui Euclid pentru determinarea celui mare divizor comun a două numere naturale nenule citite, pentru utilizarea acestuia în contextul simplificării unei fracții - aplicarea algoritmului de generare a elementelor unei liste ai cărei termeni respectă o regulă dată, în contextul determinării numărului de ani după care soldul contului bancar, având o sumă inițială și o dobândă anuală, depășește o valoare dată - implementarea algoritmului de sortare cu listă de frecvențe pentru ordonarea notelor la testul de la informatică al unei clase de elevi (note întregi între 1 și 10), cu construirea listei de frecvențe prin numărarea frecvenței fiecărei note și afișarea notelor sortate prin parcurgerea listei - aplicarea metodei bulelor pentru ordonarea tuturor secvențelor de numere pozitive dintr-o listă (de exemplu, o listă de temperaturi înregistrate într-un oraș), prin implementarea algoritmului într-un limbaj de programare |
| CS 3.3. Aplică strategii de rezolvare a problemelor prin proiectarea modulară a algoritmilor |
| <ul style="list-style-type: none"> - descompunerea unei probleme, având descrierea rezolvării acesteia, în subprobleme (module), indicând etapele de lucru pentru fiecare - utilizarea principiilor de proiectare modulară pentru o aplicație simplă (de exemplu, gestionarea unor cheltuieli și încasări lunare), atribuind câte un modul fiecărui membru al unei echipe - aplicarea unui model de proiectare modulară în realizarea unei soluții, în vederea efectuării etapelor de analiză, proiectare, testare în cadrul fiecărui modul |
| CS 3.4. Utilizează principalele elemente ale limbajului de programare utilizate pentru prelucrarea datelor organizate în modele fundamentale - date simple sau liste, respectiv pentru transmiterea datelor de la și către un program, în vederea rezolvării eficiente a problemelor informatice |
| <ul style="list-style-type: none"> - aplicarea operatorilor de concatenare (+) și multiplicare (*) în rezolvarea unei situații practice de gestionare a comenzilor din două depozite, prin combinarea listelor de produse și dublarea unui stoc existent - aplicarea metodelor <code>count()</code> și <code>index()</code> în rezolvarea unei situații practice de numărare a produselor identice și de identificare a poziției unui articol într-o listă de stocuri, prin scrierea unui program Python simplu - aplicarea metodelor <code>remove()</code> și <code>clear()</code> pentru prelucrarea datelor într-o aplicație de gestionare a unui coș de cumpărături, prin scrierea unui program Python simplu care elimină produsele selectate de utilizator și golește lista după finalizarea comenzii - completarea unui program "lacunar" care citește o valoare întreagă dintr-un fișier text pentru a scrie diverse rezultate obținute tot într-un fișier text (de exemplu pătratul și rădăcina pătrată sau divizorii acestuia) - adăugarea unei casete de text într-o aplicație simplă, dată, cu o fereastră, o casetă de text și un buton care afișează un rezultat într-o fereastră de mesaje - <code>MessageBox</code> (de exemplu numărul de divizori), astfel încât rezultatul afișat să reflecte ambele valori din casetele de text (de exemplu numărul de divizori comuni) |
| CS 3.5. Utilizează elementele de sintaxă și componentele din definiția și apelul subprogramelelor și subprogramelelor predefinite pentru operații uzuale, în implementarea algoritmilor în limbaj de programare |

| Clasa a IX-a |
|---|
| <ul style="list-style-type: none"> - folosirea conceptului de transmitere a parametrilor în rezolvarea unei situații practice de calcul al temperaturii maxime, prin apelul unui subprogram Python ce returnează maximul dintre două numere - implementarea sintaxei de definire și apel a unui subprogram Python pentru rezolvarea unei situații practice de calcul al ariei unui teren, folosind ca parametri lungimea și lățimea introduse de utilizator - implementarea subprogramelor cu funcțiile predefinite <code>abs()</code> și <code>sqrt()</code> în rezolvarea unei situații practice de calcul al distanței dintre două puncte într-un plan, folosind ca parametri coordonatele introduse de utilizator - aplicarea funcțiilor predefinite pentru colecții în rezolvarea unei situații practice de calcul al performanței sportivilor (de exemplu, determinarea mediei, a celei mai mari și celei mai mici valori a scorurilor), prin scrierea unui program Python simplu |

CG 4 - Analizează caracteristicile și aplicabilitatea modelelor conceptuale și operaționale ale dezvoltării produselor software, pentru a selecta soluțiile cele mai potrivite în funcție de contextul informatic dat

| Clasa a IX-a |
|---|
| CS 4.1. Analizează caracteristicile, modul de prelucrare, avantajele și dezavantajele utilizării unor modele conceptuale fundamentale - date simple sau liste, pentru a alege structura adecvată în rezolvarea unor probleme concrete |
| <ul style="list-style-type: none"> - analizarea diferențelor conceptuale dintre stivă, coadă și listă, prin compararea modului de inserare și extragere a elementelor și examinarea avantajelor și limitărilor fiecărui model în contexte reale, precum gestionarea apelurilor telefonice (coadă) versus procesarea comenzilor de tip „anulare ultimă acțiune” (stivă) - analizarea modului de formare și interpretare a unei liste de frecvențe în studierea obiceiurilor de cumpărare ale clienților, prin descompunerea procesului în etape (colectare, numărare, reprezentare), corelarea valorilor din listă cu comportamentele observate - analizarea modului de funcționare a algoritmilor de parcurgere liniară prin compararea etapelor de verificare, selecție și filtrare a datelor într-o situație practică precum monitorizarea tranzacțiilor zilnice ale unui magazin, pentru a evidenția ordinea logică a operațiilor și dependențele dintre ele - analizarea modului de organizare și prelucrare a unei liste de valori memorate în monitorizarea zilnică a consumului de apă al unei gospodării, prin identificarea relației dintre ordinea datelor, valorile stocate și calculele derivate, precum determinarea consumului mediu și a variațiilor semnificative |
| CS 4.2. Analizează comportamentul, aplicabilitatea, avantajele și dezavantajele utilizării unor algoritmi specializați pe clase de probleme pentru prelucrarea numerelor, sortarea sau generarea sistematică a unor secvențe de valori |
| <ul style="list-style-type: none"> - analizarea a doi algoritmi alternativi dați pentru determinarea divizorilor unui număr în vederea estimării numărului de operații efectuate - urmărirea raționamentului rezultat prin operațiile din cadrul unui algoritm pentru a verifica dacă acesta determină obținerea oglinditului unui număr natural - analizarea avantajelor utilizării divizorilor unui număr în contexte practic-aplicative (de exemplu, găsirea tuturor posibilităților pentru tăierea unei benzi de L centimetri în bucăți de lungimi egale (numere naturale), fără pierderi de material, posibilitatea de distribuire a N caiete unui număr de elevi astfel încât aceștia să primească același număr de caiete, ambalarea a N produse în cel puțin două pachete astfel încât numărul de pachete să fie minim și în fiecare pachet să fie același număr de produse) - analizarea comportamentului unui algoritm pentru formarea unei liste date (de exemplu, lista primelor n pătrate perfecte consecutive), urmărind raționamentul rezultat prin operațiile din cadrul acestuia pentru a aprecia adecvarea acestuia la cerință - analizarea algoritmului de sortare prin selecția minimului, pentru o listă cu n valori, estimând numărul de comparații pentru un caz favorabil, un caz mediu și un caz nefavorabil - compararea metodei de selecție a minimului cu altă metodă de sortare (de exemplu, metoda bulelor) aplicate pe un set de 200 de punctaje obținute de participanții la olimpiada de informatică din care se vor obține, în ordine descrescătoare, cele mai mari 10 punctaje, din perspectiva numărului de comparații și interschimbări |
| CS 4.3. Analizează aplicabilitatea, avantajele și dezavantajele utilizării unor strategii de rezolvare a problemelor prin proiectarea modulară a algoritmilor |
| <ul style="list-style-type: none"> - analizarea erorilor apărute în programele informatice în scopul încadrării lor într-una dintre categoriile: erori de sintaxă, de logică, de executare - analizarea modulelor unui cod sursă al unui sistem complex de validare a datelor (de ex., un sistem care validează formulare web), punând în evidență contribuția fiecărui modul la procesul general de validare - analizarea avantajelor și dezavantajelor utilizării modulelor în cadrul elaborării unui program - analizarea modului în care interacționează modulele unei aplicații, pe baza unei diagrame care arată componentele acesteia și modul în care interacționează |
| CS 4.4. Analizează aplicabilitatea, avantajele și dezavantajele utilizării unor elemente ale limbajului de programare, pentru a identifica soluția adecvată prelucrării datelor organizate în modele fundamentale - date simple sau liste, respectiv pentru transmiterea datelor de la și către un program, în vederea rezolvării eficiente a problemelor informatice |

| Clasa a IX-a | |
|--|--|
| <ul style="list-style-type: none"> - analizarea diferențelor de implementare în Python între atribuirea directă a listelor și copierea acestora prin metode specifice (<code>copy()</code>-copiere superficială, <code>slicing</code>-selectarea unei secvențe), în contextul unei aplicații de procesare a datelor financiare, prin observarea efectelor modificării elementelor asupra listelor asociate - analizarea efectelor utilizării metodelor <code>copy()</code> și <code>sort()</code> asupra conținutului listelor, în contextul unei aplicații de evidență a notelor elevilor, prin compararea rezultatelor obținute înainte și după sortare, respectiv copiere - analizarea diferențelor de comportament dintre metodele <code>sort()</code> și <code>reverse()</code> din Python aplicate asupra unei liste de prețuri, în contextul unei aplicații de evidență a vânzărilor, prin observarea și explicarea ordinii obținute și a modificării listei inițiale - compararea timpului de executare a unui program care citește datele dintr-un fișier în raport cu același program care citește datele de la tastatură - analizarea avantajelor și dezavantajelor organizării elementelor grafice într-o fereastră, a textelor și a mesajelor asociate, în logica "user-friendly=ușor de utilizat" | |
| CS 4.5. Analizează aplicabilitatea, avantajele și dezavantajele utilizării subprogramelor, în implementarea algoritmilor în limbaj de programare | |
| <ul style="list-style-type: none"> - analizarea descompunerii unui program complex de prelucrare a datelor în subprograme, prin identificarea variabilelor locale și globale și explicarea modului în care acestea influențează funcționarea independentă a fiecărui subprogram - analizarea diferențelor dintre apelul unui subprogram cu parametri și fără parametri, în contextul unui proiect de prelucrare a datelor meteo, prin observarea modului în care se transmit și se utilizează datele - analizarea efectului diferitelor funcții de conversie (<code>int()</code>, <code>float()</code>, <code>str()</code>) asupra acelorași date de intrare, în contextul unei aplicații de procesare a prețurilor produselor folosind subprograme, prin compararea rezultatelor și explicarea diferențelor de tip de date - analizarea modului în care rezultatele funcțiilor <code>min()</code> și <code>max()</code> pot fi influențate de structura și tipul elementelor din colecție, în contextul unei aplicații de prelucrare a prețurilor produselor folosind subprograme, prin compararea rezultatelor pentru liste eterogene | |

CG 5 - Evaluează corectitudinea și eficiența soluțiilor informatice, în vederea optimizării și asigurării funcționalității în diverse scenarii de utilizare

| Clasa a IX-a | |
|--|--|
| CS 5.1. Argumentează alegerea unor modele conceptuale fundamentale - date simple sau liste - și a modurilor de prelucrare a acestora pentru rezolvarea unor probleme informatice concrete | |
| <ul style="list-style-type: none"> - evaluarea utilizării adecvate a unei liste, stive sau cozi într-o situație concretă, cum ar fi gestionarea cererilor de suport într-un serviciu de asistență tehnică, prin argumentarea alegerii structurii potrivite în funcție de ordinea de procesare, prioritate și tipul de acces necesar - evaluarea relevanței și acurateței unei liste de frecvențe în prezentarea rezultatelor unei anchete de opinie, prin verificarea modului în care datele reflectă realitatea studiată, argumentarea corectitudinii structurii listei și formularea unei concluzii privind utilitatea sa pentru luarea deciziilor - alegerea unui algoritm de parcurgere liniară fără memorare pentru procesarea datelor provenite dintr-un registru de intrări, prin aprecierea clarității pașilor logici, a modului de acces la date și argumentarea deciziei privind eficiența și corectitudinea abordării alese - evaluarea corectitudinii și relevanței memorării datelor organizate într-o listă utilizată pentru înregistrarea și compararea vânzărilor lunare ale unui produs, prin verificarea completitudinii datelor stocate, a ordinii lor și a modului de interpretare în raport cu scopul analizei comerciale | |
| CS 5.2. Evaluează corectitudinea și eficiența soluțiilor cu algoritmi specializați pe clase de probleme pentru prelucrarea numerelor, sortarea sau generarea sistematică a unor secvențe de valori | |
| <ul style="list-style-type: none"> - evaluarea complexității a doi algoritmi dați pentru determinarea divizorilor unui număr natural stabilind care dintre cei doi algoritmi este mai eficient (de exemplu, parcurgere a posibilibilor divizori de la 1 până la n, până la $n/2$ sau până la \sqrt{n}) - argumentarea faptului că un algoritm ce construiește oglinditul unui număr și apoi oglinditul oglinditului nu conduce totdeauna la obținerea numărului inițial, susținută de teste adecvate și compararea rezultatelor obținute cu cele așteptate - justificarea folosirii unui algoritm de generare recurentă a termenilor unei expresii matematice pentru însumarea unor produse, în comparație cu folosirea unui algoritm de calcul direct al fiecărui termen al acesteia (de exemplu pentru a calcula suma $1+1\cdot2+1\cdot2\cdot3+\dots+1\cdot2\cdot3\cdot\dots\cdot n$) - evaluarea eficienței (calculând ordinul de complexitate) unui algoritm de obținere a termenilor șirului lui Fibonacci, pornind de la un termen dat, în ordine descrescătoare, prin două metode: prima metodă memorează ultimul termen afișat apoi îl determină pe cel curent, care îl precede în șir, prin generarea, începând de la 1, în ordine crescătoare, a tuturor termenilor, până la cel căutat, iar a doua metodă generează toți termenii în ordine crescătoare, o singură dată, începând de la 1, până la termenul care îl precede pe cel dat inițial, apoi generarea fiecărui termen curent se face prin diferență, pe baza adaptării regulii de generare specifice - evaluarea corectitudinii unui algoritm care generează o listă cu primii termeni ai șirului lui Fibonacci, pe baza unor teste adecvate care au în vedere și cazuri limită, de exemplu listă vidă, listă cu un singur element, listă cu doar două elemente | |

| Clasa a IX-a |
|--|
| <ul style="list-style-type: none"> - argumentarea deciziei de utilizare a metodei de sortare cu listă de frecvențe pentru un sistem de triaj al pacienților din camera de urgență a unui spital, sistem bazat pe ordonarea descrescătoare a gravității stării lor medicale (codificată prin valori de la 1 la 5), ținând cont de caracteristicile datelor (număr mare de elemente, interval numeric limitat) |
| CS 5.3. Evaluează corectitudinea și eficiența algoritmilor care utilizează strategii de rezolvare a problemelor prin proiectarea modulară a algoritmilor |
| <ul style="list-style-type: none"> - argumentarea avantajelor descompunerii unei probleme complexe în etape sau subprobleme mai simple (citire date, validare, rezolvare propriu-zisă, afișare rezultat) din perspectiva verificării corectitudinii fiecărui modul - evaluarea timpului de executare a diferiților algoritmi pentru seturi de date de dimensiuni diferite și reprezentarea grafică a rezultatelor - evaluarea comparativă a două soluții ale unei probleme, una modulară și una compactă, argumentând care este mai eficientă - evaluarea corectitudinii și eficienței unei soluții propuse de colegi, oferind feedback privind claritatea și independența modulelor |
| CS 5.4. Evaluează corectitudinea și eficiența aplicațiilor care utilizează elemente specifice de limbaj de programare pentru prelucrarea datelor organizate în modele fundamentale - date simple sau liste, respectiv pentru transmiterea datelor de la și către un program, în vederea rezolvării eficiente a problemelor informatice |
| <ul style="list-style-type: none"> - justificarea corectitudinii și eficienței utilizării operatorilor de apartenență (in, not in) într-un program de validare a datelor introduse de utilizator, în cadrul unei dezbateri despre criterii de claritate și siguranță în prelucrarea listelor - evaluarea unui program care utilizează obiecte din clasa list, și determină numărul de perechi de valori pare, aflate pe poziții consecutive în listă, pe baza unor teste adecvate care au în vedere comportamentul acestuia pentru elemente care au câte doi vecini, sau se află la fiecare dintre cele două extremități ale listei - evaluarea corectitudinii unui program cu fișiere text, pe baza unor teste adecvate care au în vedere și cazuri limită, de exemplu fișier inexistent, fișier gol - compararea unei aplicații cu interfață grafică în raport cu aceeași aplicație cu interfață text, evidențiind valențele fiecăreia (avantaje și dezavantaje) |
| CS 5.5. Evaluează corectitudinea și eficiența programelor care utilizează subprograme, pentru a rezolva probleme informatice |
| <ul style="list-style-type: none"> - susținerea cu argumente a celor două variante de implementare a aceluiași algoritm (una cu subprograme, alta fără), în cadrul unei dezbateri despre eficiența, claritatea și reutilizarea codului - justificarea alegerii unei variante de definire și apel al unui subprogram Python, din mai multe variante posibile, într-un proiect de calcul al salariului net, prin aplicarea unor criterii de lizibilitate, modularitate și reutilizare a codului - evaluarea acurateții și relevanței utilizării funcțiilor de rotunjire (round(), math.floor(), math.ceil()) într-o aplicație de calcul al totalului facturii, în contextul unei discuții despre criterii de precizie financiară - justificarea utilizării funcției sum() pentru calculul sumei elementelor dintr-o colecție, în comparație cu iterarea explicită, evidențiind avantajele din perspectiva clarității codului și reducerii erorilor, în contextul unei discuții despre bune practici de programare și optimizare a codului - evaluarea corectitudinii unei aplicații cu subprograme, pe baza unor teste adecvate care au în vedere și cazuri limită, de exemplu pentru parametri cu valori inadecvate, lipsa unei valori returnate pentru unele cazuri |

CG 6 - Elaborează algoritmi și programe personalizate, pentru a crea soluții informatice coerente și adaptate cerințelor

| Clasa a IX-a |
|---|
| CS 6.1. Proiectează algoritmi personalizați care utilizează modele conceptuale fundamentale - date simple sau liste pentru organizarea și prelucrarea eficientă a datelor, utilizând algoritmi de bază, în vederea rezolvării unor probleme |
| <ul style="list-style-type: none"> - proiectarea unui model conceptual care combină principiile listelor, stivelor și cozilor pentru optimizarea fluxului de sarcini într-un centru de livrări, printr-un sistem de organizare a comenzilor care stochează, prioritizează și eliberează elementele conform regulilor de acces definite pentru fiecare tip de structură - proiectarea unui model conceptual de listă de frecvențe pentru monitorizarea numărului de erori raportate zilnic, pentru fiecare tip posibil, într-un centru de asistență tehnică, prin stabilirea structurii listei, a modului de înregistrare a fiecărei erori și a regulilor de actualizare a frecvențelor, astfel încât modelul să poată fi ulterior implementat în limbaj de programare - proiectarea unui algoritm de parcurgere liniară pentru determinarea și prelucrarea automată a valorilor neconforme dintr-o listă de date financiare, prin stabilirea pașilor de acces succesiv la elemente, definirea condițiilor de verificare și formularea logicii generale de procesare, care va constitui ulterior baza implementării în limbaj de programare - proiectarea unei liste de valori memorate pentru urmărirea evoluției zilnice a nivelului de poluare într-un oraș, prin definirea modului de colectare și stocare a valorilor unui anumit tip de noxe, stabilirea algoritmilor de parcurgere pentru calculul valorii medii și identificarea zilelor critice cu valori care depășesc media, ca bază pentru o viitoare implementare în limbaj de programare |

| Clasa a IX-a | |
|--|--|
| CS 6.2. Proiectează soluții cu aplicarea personalizată a algoritmilor specializați pe clase de probleme pentru prelucrarea numerelor, sortarea sau generarea sistematică a unor secvențe de valori | |
| <ul style="list-style-type: none"> - proiectarea unui program ce implementează operații cu câte două fracții date prin numărătorul și numitorul lor (simplificare, adunare, scădere, înmulțire, împărțire), utilizând personalizat algoritmi de divizibilitate - rezolvarea unei probleme ce determină cel mai mic multiplu comun a două numere pe baza celui mai mare divizor comun al lor, pentru obținerea duratei minime de timp după care două semafoare trec simultan la aceeași culoare, cunoscându-se, pentru fiecare, timpul de menținere a fiecărei culori (roșu și verde) - conceperea unui algoritm care determină numărul de persoane care sunt născute într-o anumită lună a anului pe baza unei liste de coduri numerice personale date - crearea unui algoritm eficient care afișează primii n termeni ai unui șir recurent, prin determinarea formulei termenului general, cunoscându-se primii termeni și relația de recurență (de exemplu, pentru $a_0=3$ și relația de recurență $a_n=2 \cdot a_{n-1}$, pentru $n>0$, se obține $a_n=3 \cdot 2^n$) - implementarea unui program pentru gestionarea solicitărilor de mentenanță pentru o companie de lifturi, care își sortează cererile în funcție de tipul defecțiunii (codificat cu valori de la 1 la 8), obținându-se în final lista cererilor ordonate - dezvoltarea unui program care utilizează metoda bulelor pentru organizarea programărilor zilnice ale unei clinici medicale în ordinea crescătoare a duratei consultațiilor, prin proiectarea algoritmului, testarea funcționării sale pe un set de date simulate și realizarea unei afișări care să evidențieze clar modul de aplicare al metodei | |
| CS 6.3. Proiectează modular algoritmi personalizați pentru rezolvarea problemelor | |
| <ul style="list-style-type: none"> - descompunerea unei aplicații în module, în cadrul unui grup, în care fiecare modul este abordat de câte un membru al grupului, și integrarea acestor rezolvări, ulterior, într-un produs unitar - conceperea unei diagrame generale pentru rezolvarea modulară a unei probleme, evidențiind componentele și conexiunile dintre ele - completarea unei aplicații existente (de exemplu: o aplicație de gestionare a bibliotecii școlare, un sistem de evidență a prezenței, un registru digital etc.), a cărei descriere este dată, cu un modul nou, care adaugă o funcționalitate suplimentară (de exemplu, raportare, notificări, export de date, interfață de administrare etc.), respectă principiile de proiectare modulară și poate fi integrat în structura existentă | |
| CS 6.4. Implementează aplicații care integrează în mod personalizat elemente specifice de limbaj de programare pentru prelucrarea datelor organizate în modele fundamentale - date simple sau liste, respectiv pentru transmiterea datelor de la și către un program, în vederea realizării unor soluții funcționale și performante | |
| <ul style="list-style-type: none"> - elaborarea unei aplicații Python de gestiune a listelor de produse dintr-un magazin online, prin integrarea operatorilor de acces, apartenență, concatenare și relaționare pentru actualizarea și compararea automată a stocurilor din mai multe surse - proiectarea unei aplicații Python pentru administrarea unei liste de produse dintr-un magazin, utilizând metodele clasei list (<code>append()</code>, <code>insert()</code>, <code>pop()</code>, <code>count()</code>, <code>index()</code>, <code>copy()</code>, <code>sort()</code>) pentru a adăuga, elimina, sorta și analiza elementele din stoc - elaborarea unei aplicații Python pentru gestionarea unei liste dinamice de sarcini (to-do list), prin identificarea și utilizarea metodelor adecvate (<code>append()</code>, <code>remove()</code>, <code>sort()</code>, <code>reverse()</code>, <code>copy()</code>, <code>clear()</code>) pentru adăugarea, eliminarea, ordonarea și resetarea elementelor - proiectarea unei aplicații de exploatare cu valențe practice a unui fișier text (de generare a unor serii aleatoare de numere folosind biblioteca random urmată de numărări și rezultate statistice pentru o astfel de serie generată) - implementarea unei aplicații Python pentru afișarea, într-o fereastră, cu fundal colorat, a câte unui termen aparținând unui șir generat pe baza unei relații de recurență, utilizând două obiecte din clasa Button, cu proprietatea text completată sugestiv, pentru a marca pornirea, respectiv oprirea procesului de obținere a termenilor șirului | |
| CS 6.5. Implementează aplicații în limbaj de programare care integrează subprograme personalizate, pentru a modulariza și organiza eficient codul în cadrul soluțiilor informatice | |
| <ul style="list-style-type: none"> - construirea unui mini-proiect de tip aplicație modulară Python (de exemplu, gestionarea unui catalog școlar), prin proiectarea și implementarea propriilor subprograme care utilizează parametri și returnează valori, cu prezentarea funcționalității în fața clasei - proiectarea unei aplicații modulare Python de tip calculator personalizat (ex. conversii valutare, conversii de unități), prin definirea și apelarea mai multor subprograme proprii, utilizând parametri și valori returnate relevante pentru funcționalitatea dorită - construirea unui mini-proiect de tip aplicație modulară Python pentru conversia și prelucrarea valorilor numerice (de exemplu, o aplicație care convertește temperaturi, distanțe sau sume de bani), prin integrarea mai multor funcții predefinite de calcul și conversie (<code>abs()</code>, <code>round()</code>, <code>int()</code>, <code>float()</code>, <code>sqrt()</code>) - proiectarea unei aplicații modulare Python care gestionează o colecție de date reale (de exemplu, temperaturi zilnice, vânzări lunare, note școlare), utilizând funcțiile <code>len()</code>, <code>min()</code>, <code>max()</code>, <code>sum()</code> pentru generarea unui raport sintetic cu informații despre datele analizate | |
| CONȚINUTURI ALE ÎNVĂȚĂRII | |

Clasa a IX-a

| Domenii de conținut | Conținuturi |
|---|---|
| 1. Organizarea conceptuală a datelor | <p>1.1. Modelul conceptual liniar - listă</p> <ul style="list-style-type: none"> - caracteristici, din punctul de vedere conceptual, ale unei liste și ale cazurilor particulare date de tipul de acces (stivă, coadă, respectiv cu acces direct, cu acces secvențial), rolul avut în prelucrarea datelor (listă de frecvențe); - repere pentru parcurgerea elementelor și pentru aplicarea algoritmilor de bază pentru prelucrarea datelor organizate liniar, cu sau fără memorare. |
| 2. Strategii de rezolvare a problemelor | <p>2.1. Principii de elaborare a unui program</p> <ul style="list-style-type: none"> - caracteristici ale gândirii computaționale și ale principalelor etape ale elaborării unui program (analiză, proiectare, implementare, testare și depanare), repere pentru aplicarea acestor etape în rezolvarea problemelor; - caracteristici de bază, avantaje și limite ale principalelor moduri de reprezentare a algoritmilor: blocuri grafice, pseudocod, limbaj de programare (cu interpretor sau compilator); - repere pentru proiectarea modulară a algoritmilor; - criterii de elaborare a testelor (pentru validarea datelor și pentru verificarea comportamentului algoritmului), repere pentru urmărirea evoluției valorilor variabilelor în scopul identificării și tratării erorilor; - eficiența algoritmilor: caracteristici ale eficienței din punctul de vedere al spațiului de memorie și din punctul de vedere al timpului de executare, criterii pentru determinarea ordinului de complexitate a unui algoritm polinomial (notația O); - caracteristici ale unor modalități de bază de comunicare cu programul: interfață de tip consolă, interfață grafică (ferestre, butoane, etichete, casete text), fișiere. <p>2.2. Prelucrări ale datelor numerice</p> <ul style="list-style-type: none"> - operații specifice cu cifrele unui număr (acces la o cifră a unui număr, adăugare a unei cifre la stânga unui număr, adăugare a unei cifre la dreapta unui număr), determinarea unui divizor/multiplu al unui număr; - repere pentru prelucrarea numerelor (de exemplu parcurgerea cifrelor unui număr, a divizorilor unui număr, descompunere a unui număr în factori primi) și aplicarea algoritmilor de bază; - algoritmi elementari pentru determinarea celui mai mare divizor comun (algoritmul lui Euclid cu scăderi repetate, respectiv cu împărțiri repetate); - repere pentru transformarea unui număr dintr-o bază de numerație în altă bază de numerație; <p>2.3. Metode de generare sistematică a elementelor unei liste</p> <ul style="list-style-type: none"> - repere pentru generarea unor secvențe de valori (de exemplu secvențe cu proprietăți date, termeni ai unor expresii matematice, termeni ai unor șiruri recurente) și aplicarea algoritmilor de bază. <p>2.4. Metode de sortare a elementelor unei liste</p> <ul style="list-style-type: none"> - caracteristici ale metodei de sortare prin selecția minimului și repere pentru aplicarea metodei; - caracteristici ale metodei de sortare cu listă de frecvențe și repere pentru aplicarea metodei; - caracteristici ale metodei bulelor și repere pentru aplicarea metodei. |
| 3. Memorarea datelor și organizarea codului în limbaj de programare | <p>3.1. Subprograme</p> <ul style="list-style-type: none"> - caracteristici, rol; - concepte de bază și caracteristici: antet, corp, variabile locale, variabile globale, transmitere prin intermediul parametrilor, returnare a rezultatelor, apel, mecanism de executare; - sintaxă pentru definiția și apelul unui subprogram în Python; - caracteristici și repere de utilizare a unor subprograme predefinite în Python pentru operații matematice uzuale (de exemplu radical, valoare absolută, parte întreagă, rotunjire) și conversii; - caracteristici și repere de utilizare a unor subprograme predefinite în Python pentru colecții de valori (determinarea numărului de elemente, a minimului, maximumului, sumei unor valori). <p>3.2. Introducere în programarea orientată pe obiecte în limbaj de programare</p> <ul style="list-style-type: none"> - noțiuni de bază necesare utilizării unor clase predefinite: clasă, membri ai clasei (date și metode), obiecte, biblioteci; - instanțiere a unei clase predefinite, acces la membrii unui obiect. <p>3.3. Fișiere text</p> |

| Domenii de conținut | Conținuturi |
|---------------------|--|
| | <ul style="list-style-type: none"> - caracteristici, principii de lucru (deschidere, închidere, transfer de date); - funcție pentru deschiderea unui fișier în Python, clasa predefinită TextIOWrapper din Python pentru prelucrarea fișierelor: caracteristici, metode de bază (pentru citire, scriere, închidere); - repere pentru identificarea și utilizarea altor clase și metode, adecvate pentru prelucrarea fișierelor și pregătirea datelor citite din fișiere. <p>3.4. Biblioteca Tkinter din Python pentru interfețe grafice</p> <ul style="list-style-type: none"> - clase, caracteristici, funcții și metode de bază pentru interfețe grafice simple cu ferestre, butoane, etichete, casete text (Tk, Label, Button, Entry, Text, Frame, Canvas), afișare a mesajelor, comportament (pack, grid, place, get); - repere pentru identificarea și utilizarea altor clase și metode pentru realizarea interfețelor grafice. <p>3.5. Clasa list din Python – clasă predefinită pentru memorarea unei liste</p> <ul style="list-style-type: none"> - caracteristici; - operatori specifici pentru acces la un element, apartenență, non-apartenență, concatenare, multiplicare, relaționare; - metode ale clasei pentru operații de bază: determinare a primei poziții a unei valori, numărare a aparițiilor unei valori, ștergere a elementului de la o anumită poziție, inserare a unei valori într-o anumită poziție, adăugare a unui element, copiere, sortare; - repere pentru identificarea și utilizarea altor metode ale clasei, adecvate pentru prelucrarea listelor. |

SUGESTII METODOLOGICE

Sugestiile metodologice au rolul de a sprijini profesorii în aplicarea programei, fără a impune metode unice sau rigide. Acestea traduc intențiile programei (competențe generale, competențe specifice, conținuturi, exemple de activități de învățare) în moduri concrete de lucru la clasă și oferă repere pentru organizarea învățării și privind evaluarea rezultatelor învățării, pentru alegerea strategiilor didactice și pentru integrarea conținuturilor și competențelor în practica școlară. Această secțiune are caracter aplicativ, nu teoretic: nu inventariază metode, strategii sau instrumente, ci oferă exemple minimale, relevante.

Disciplina *informatică* are atât caracter teoretic, academic, cât și practic, iar activitățile din cadrul instruirii teoretice se desfășoară, de regulă, în săli de clasă, dotate cu tablă interactivă, pentru exemplificarea programelor pe calculator, iar activitățile din cadrul instruirii practice se desfășoară în laboratorul de informatică, unde este indicat ca fiecare elev să dispună de un calculator propriu, conectat la rețea și la Internet, pentru a facilita formarea competențelor prevăzute în programă. Stațiile de lucru trebuie să fie configurate astfel încât să permită executarea aplicațiilor specifice, iar organizarea calculatoarelor să fie plasate în formă de U sau cu o orientare către tabla principală, pentru o vizibilitate optimă.

Principiile generale care trebuie să guverneze activitatea de predare-învățare-evaluare cuprind:

- centrare pe elev: strategiile să favorizeze implicarea activă a elevilor, de exemplu învățarea prin descoperire, colaborarea și reflecția personală;
- diversitate metodologică: se recomandă utilizarea de metode variate;
- flexibilitate: profesorii adaptează activitățile de învățare la nivelul clasei și la resursele disponibile;
- corelare cu profilul de formare al absolventului: metodele didactice trebuie alese astfel încât să contribuie la formarea competențelor-cheie și a atributelor prioritare ale absolventului;
- integrare interdisciplinară: învățarea devine mai relevantă atunci când disciplinele se sprijină reciproc și creează punți între conținuturi;
- îmbinarea evaluării formative cu cea sumativă, cu recomandarea unor strategii de evaluare centrate pe o reflecție profundă asupra întrebărilor esențiale precum: De ce evaluez? Ce evaluez? Cum evaluez? Cât de bine măsoară? Ce feedback dau? Ce decizii iau?;
- diferențiere/ personalizare: adaptarea parcursului didactic la situații specifice (de exemplu: elevi cu CES și/ sau dizabilități, elevi cu ritm înalt de învățare, elevi care au nevoie de învățare remedială, elevi în risc de abandon etc.).

Orientările metodologice generale cuprind:

- învățare activă: dezbateri, studii de caz, proiecte, portofolii, simulări, investigații;
- învățare colaborativă: activități în grup, peer-to-peer, mentorat între elevi;
- învățare prin proiect: integrarea conținuturilor disciplinei în teme mai largi (sociale, științifice, culturale);
- învățare cu suport digital: utilizarea resurselor online, aplicații interactive, simulări virtuale;
- învățare autentică: activități conectate cu realitatea cotidiană și cu problemele comunității;
- învățare contextualizată: activități corelate cu specificul elevilor din clasă/școala în care predă profesorul.

Se recomandă adaptări metodologice după tipul de programă, de exemplu, pentru disciplinele din curriculumul de specialitate:

- accent pe elemente de specializare, pe aprofundare, pe rezolvarea de probleme complexe și pe gândire critică;
- metode exploratorii, investigații, cercetări aplicate;
- exemple: proiecte interdisciplinare, aplicații în domenii profesionale, utilizarea software-urilor specializate.

Formarea competențelor trebuie să aibă în vedere și legătura cu profilul de formare al absolventului, astfel încât disciplina să contribuie la dezvoltarea/consolidarea/ diversificarea:

- competențelor-cheie (ex.: competențe matematice, digitale, sociale și civice, a învăța să înveți);
- atributelor prioritare ale profilului absolventului (ex.: reflexiv, creativ, responsabil, comunicativ);
- temelor transversale prevăzute de Legea 198/2023, art. 88 alin. 10 (ex.: educație pentru mediu, digitalizare, sănătate, patrimoniu).

Activitățile de învățare trebuie să fie alese adecvat, pentru a contribui la formarea competențelor specifice din programă, astfel încât pentru nivelurile cognitive de recunoaștere și înțelegere se recomandă activități demonstrative și exerciții de identificare, pentru nivelul de aplicare se recomandă activități practice și aplicații asistate digital, pentru nivelurile de analiză și evaluare se recomandă studii de caz și proiecte interdisciplinare, iar pentru nivelul de creare se recomandă activități de tip învățare prin acțiune, realizarea de produse digitale și proiecte în echipă.

Activitățile pe calculator sunt coordonate de profesor, care definește clar sarcinile, timpul alocat și criteriile de evaluare, adaptând nivelul de dificultate în funcție de particularitățile colectivului de elevi.

Se recomandă îmbinarea metodelor didactice tradiționale de predare-învățare-evaluare (de exemplu demonstrația, problematizarea, algoritizarea, proba practică) cu cele moderne (de exemplu învățarea prin descoperire, proiectul, portofoliul), pentru a stimula gândirea computațională și autonomia elevilor în rezolvarea de probleme informatice, profesorul având un rol preponderent de îndrumare a elevilor, în loc de unul de furnizor de informații.

Mijloacele de învățământ utilizate pot fi variate, beneficiind de tehnologiile moderne care facilitează învățarea, cum ar fi aplicații specializate, software-uri educaționale, simulatoare ale comportamentului datelor prelucrate de algoritmi, tutoriale, resurse online, platforme care permit evaluarea automată a soluțiilor informatice.

Evaluarea în cadrul disciplinei informatică trebuie să aibă un caracter formativ/pe parcurs, urmărind nu doar obținerea unui rezultat corect, ci și modul în care elevii își formează gândirea algoritmică, formulează strategii, își testează algoritmi și corectează propriile erori. Se recomandă evaluări sumative/finale, după fiecare unitate de învățare.

Conform sugestiilor metodologice din programa școlară de *informatică și TIC* pentru gimnaziu, la clasele a VII-a și a VIII-a pentru formarea competențelor specifice s-a putut utiliza un limbaj de programare dintr-o listă care cuprinde, de exemplu, propuneri ca Python, C, C++. Deoarece alegerea limbajului de programare a fost la latitudinea profesorului, este necesară **o armonizare a acestor limbaje de programare la trecerea în clasa a IX-a, pentru ca toți elevii să utilizeze limbajul de programare precizat în programa școlară în vigoare pentru această clasă**. Astfel, se recomandă prezentarea, la început, a unor elemente de corespondență a principalelor instrumente de reprezentare a algoritmilor în pseudocod, blocuri grafice, limbaje de programare Python și C++: citire și afișare a datelor, biblioteci, comenzi și scripturi, instrucțiuni, variabile și tipuri de date de bază, operatori de bază, medii integrate pentru dezvoltarea programelor (IDE) și funcționalități ale acestora.

Programa disciplinei informatică are în vedere conținuturi grupate în domenii specifice, de exemplu:

1. Modelele conceptuale de organizare a datelor, care vizează reprezentări abstracte ale modului în care sunt structurate și relaționate datele într-un sistem informatic — înainte ca ele să fie memorate efectiv printr-un program sau o bază de date. Ele răspund la întrebarea: „Ce informații trebuie să prelucrez/stochez și cum sunt legate între ele?” și nu la întrebarea „Ce tip de variabile utilizez pentru a le stoca concret în memorie?”. Pe parcursul studiului disciplinei în ciclul liceal sunt studiate progresiv, din punctul de vedere al complexității relațiilor dintre date, modele conceptuale fundamentale - date simple sau liste, modele conceptuale simple - liniare, neliniare sau asociative, modele conceptuale complexe - liniare, relaționale sau ierarhice, respectiv modele conceptuale avansate - pentru proiectarea bazelor de date sau învățare automată.

2. Strategii pentru rezolvarea problemelor, care cuprind

- algoritmi specializați pe clase de probleme, cum ar fi pentru prelucrarea algoritmică a numerelor, sortarea sau generarea sistematică a unor secvențe de valori, pentru prelucrarea listelor ordonate, criptarea sau decriptarea șirurilor de caractere, pentru prelucrarea grafurilor, arborilor, algoritmi utilizați în învățarea automată;

- strategii generale de programare/abordare cum ar fi proiectarea modulară a algoritmilor, metodele de programare Greedy, Divide et impera, Backtracking și programare dinamică, normalizare a modelului conceptual al unei probleme de gestiune.

3. Elemente ale limbajului de programare pentru memorarea și prelucrarea datelor organizate în diferite modele, respectiv pentru organizarea codului (subprograme, programare orientată pe obiecte) și prelucrarea bazelor de date.

Pentru fiecare clasă, tematica precizată trebuie abordată într-o ordine logică, facilitând formarea competențelor specifice din programă, utilizând competențele și conținuturile studiate anterior, începând cu elementele de limbaj și algoritmii de bază (pentru numărare, sumă, produs, minim, maxim, prima sau ultima valoare cu o anumită proprietate, verificarea unor proprietăți) studiate la gimnaziu.

Debutul poate fi făcut, după caz, cu strategiile generale de rezolvare a problemelor, conform programei, dacă acestea pot fi aplicate în continuare, împreună cu celelalte conținuturi din cadrul anului de studiu (de exemplu, metoda Backtracking, respectiv metoda Programării dinamice sunt utilizate pentru implementarea unor algoritmi specifici grafurilor).

Pentru prelucrarea datelor organizate sub forma diferitelor modele conceptuale, se recomandă mai întâi prezentarea unor concepte de bază privind acest tip de organizare, apoi elemente de limbaj care să sprijine memorarea datelor astfel organizate, urmate de aplicarea algoritmilor de bază pentru prelucrare, respectiv de metode/algoritmii specializați pe clase de probleme pentru prelucrarea unor astfel de date (de exemplu, concepte de bază privind modelul conceptual liniar, apoi clasa list, urmată de aplicarea algoritmilor de bază pentru prelucrarea listelor, respectiv de metodele de sortare a unei liste).

Un exemplu, orientativ, de ordine de abordare a conținuturilor pentru clasa a IX-a, specializarea matematică-informatică:

1. *Strategii de rezolvare a problemelor. Principii de elaborare a unui program*
2. *Strategii de rezolvare a problemelor. Prelucrări ale datelor numerice*
3. *Memorarea datelor și organizarea codului în limbaj de programare. Subprograme*
4. *Memorarea datelor și organizarea codului în limbaj de programare. Introducere în programarea orientată pe obiecte în limbaj de programare*
5. *Memorarea datelor și organizarea codului în limbaj de programare. Biblioteca Tkinter din Python pentru interfețe grafice*
6. *Memorarea datelor și organizarea codului în limbaj de programare. Fișiere text*
7. *Organizarea conceptuală a datelor. Modelul conceptual liniar – listă*
8. *Memorarea datelor și organizarea codului în limbaj de programare. Clasa list din Python*
9. *Strategii de rezolvare a problemelor. Metode de generare sistematică a elementelor unei liste*
10. *Strategii de rezolvare a problemelor. Metode de sortare a elementelor unei liste*

Se recomandă ca în cadrul temei *Strategii de rezolvare a problemelor. Principii de elaborare a unui program* să se prezinte **doar succint** unele moduri de reprezentare a algoritmilor (blocuri grafice, schemă logică), punând în evidență utilizarea lor pe scară largă în diferite domenii, urmând ca ulterior să fie pus accentul pe pseudocod și limbajele de programare precizate în programă (în funcție de nivel și specializare), în etapa de proiectare, respectiv de implementare. Astfel, la rezolvarea **fiecărei probleme** de natură algoritmică, pe parcursul studiului disciplinei, se evidențiază aspecte specifice etapelor de elaborare a programelor:

- analiză (identificare a datelor de intrare și de ieșire, organizare conceptuală a datelor, descompunere a unei probleme în subprobleme, identificare a unor modele repetitive, abstractizare);
- proiectare (descompunere în module, reprezentare ca schemă logică, pseudocod);
- implementare (scriere a codului în limbaj de programare);
- testare (criterii de elaborare a testelor pentru validarea datelor și pentru verificarea comportamentului programului, urmărire a evoluției valorilor variabilelor pentru identificarea și tratarea eventualelor erori).

În parcurgerea conținuturilor se recomandă rezolvarea unor probleme concrete, întâlnite în viața reală, pentru a evidenția succesiunea etapelor de elaborare a unui program și pentru a dezvolta gândirea algoritmică a elevilor. Profesorul poate alterna reprezentările grafice, textuale și codificate ale aceluiași algoritm pentru a facilita înțelegerea legăturii dintre abstractizare și implementare.

În ceea ce privește prelucrarea numerelor se recomandă utilizarea exemplelor numerice și a exercițiilor practice pentru explorarea operațiilor cu cifrele unui număr, evidențiind logica accesului la acestea și compunerii numerelor. Identificarea divizorilor, descompunerea în factori primi se va realiza cu accent pe înțelegerea logicii pas cu pas, nu pe calcule complicate. Algoritmii pentru determinarea celui mai mare divizor comun vor fi prezentați intuitiv, de exemplu prin observarea împărțirilor succesive, fără formalizare matematică excesivă.

Se recomandă introducerea noțiunii de subprogram pornind de la analogii din viața reală („o rețetă”, „o instrucțiune care poate fi reutilizată”) pentru a înțelege ideea de modularitate.

Elevii pot experimenta definirea și apelul de funcții în Python prin exemple practice (calculul unei medii, determinarea unei valori maxime), iar funcțiile predefinite vor fi prezentate ca instrumente utile pentru scurtarea și claritatea codului.

Programarea orientată pe obiecte se poate introduce la clasa a IX-a prin analogii intuitive cu lumea reală, evidențiind faptul că orice obiect are proprietăți (date) și comportamente (metode), pentru a facilita înțelegerea conceptelor de clasă și membri ai clasei. Predarea are în vedere doar noțiuni **strict necesare pentru utilizarea, la nivel elementar, a unor clase predefinite**.

Profesorul poate exemplifica trecerea de la interfețele de tip consolă la cele grafice prin construirea unor mini-aplicații interactive. Crearea interfețelor grafice poate fi introdusă prin proiecte simple și atractive (de exemplu: calculator, aplicație de notițe, joc de tip „quiz”). Activitățile de învățare pot urmări legătura dintre interacțiunea elevului cu programul și comportamentul acestuia, stimulând interesul pentru proiectare și creativitate.

Lucrul cu fișiere text se poate aborda prin activități practice de învățare care ilustrează procesul complet de deschidere, citire, scriere și închidere a unui fișier, folosind exemple apropiate de experiența elevilor (liste de elevi, jurnale, notițe).




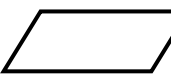
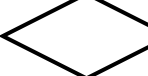


În baza modelului conceptual liniar de organizare a datelor, listele sunt privite ca **o succesiune de date, care vor fi sau nu memorate**; acestea nu sunt abordate ca liste înlanțuite la clasa a IX-a.

Modelul conceptual de organizare a datelor de tip listă poate fi introdus pornind de la contexte familiare (de exemplu: liste de cumpărături, clasamente, cozi de așteptare), pentru a facilita înțelegerea ideii de organizare și acces la date. Stiva și coada pot fi explicate intuitiv prin activități practice (de exemplu aranjarea unor obiecte fizice) pentru evidențierea regulilor de acces la elemente, adăugare și eliminare a acestora. Parcurgerea și prelucrarea elementelor se pot exercita prin activități concrete: numărarea elementelor, identificarea pozițiilor, ordonarea după criterii simple; algoritmii de bază pot fi aplicați inițial pe liste cu număr mic de elemente, pentru înțelegerea logicii, nu pentru complexitate.

Elevii pot explora metodele și proprietățile clasei list din Python prin activități interactive care simulează manipularea unor colecții de obiecte (adăugare, ștergere, sortare). Se recomandă utilizarea metodelor uzuale (de exemplu append, remove, sort) în contexte concrete, punând accent pe experimentare și descoperire a efectului comenzii. Exemplele vor fi scurte, intuitive, centrate pe calcule matematice simple.

Pentru algoritmii de sortare prin selecția minimului, listă de frecvențe și metoda bulelor se recomandă utilizarea animațiilor digitale sau simulărilor cu scopul de a vizualiza modul de funcționare pas cu pas al algoritmilor, însoțite de discuții despre situații practice unde este necesară ordonarea.

Pentru reprezentarea schemelor logice, simbolurile uzuale sunt:

| Simbol | |
|---|--|
|  | Linie de flux: reprezintă trecerea controlului între formele conectate |
|  | Bloc de proces: reprezintă un bloc de calcul |
|  | Bloc de subprogram/modul: reprezintă un apel de subprogram/modul care este reprezentat printr-o schemă logică separată |
|  | Bloc de intrare/ieșire: reprezintă intrarea (CITIRE) sau ieșirea (SCRIERE) datelor |
|  | Bloc de decizie: reprezintă o decizie care are ca rezultat două ieșiri, reprezentând cele două valori posibile (Da/Nu sau Adevărat/Fals) |
|  | Bloc terminal: reprezintă momentul de început (START) sau de final (STOP) al procesului |
|  | Bloc conector: reprezintă conectarea a două sau mai multe puncte ale schemei logice |

Pentru reprezentarea algoritmilor în pseudocod, convențiile uzuale sunt:

Indentare: liniile sunt indentate pentru a indica faptul că sunt conținute într-o structură dintr-o linie anterioară

Operațiile de intrare și ieșire sunt precizate prin comenzile **citește** <identificator sau identificatori, separați prin virgulă> respectiv **scrie** <expresie sau expresii, separate prin virgulă>

Operațiile de atribuire sunt precizate în următorul format: <identificator> ← <expresie>

Structurile de control sunt precizate în următoarele formate:

- structură de decizie

| | | |
|--|-----|---|
| <pre> dacă <condiție> atunci <Instrucțiuni> [altfel <Instrucțiuni>] </pre> | sau | <pre> dacă <condiție> atunci <Instrucțiuni> [altfel <Instrucțiuni>] sfârșit dacă </pre> |
|--|-----|---|

- structură repetitivă cu test inițial

| | | |
|---|-----|--|
| <pre> cât timp <condiție> execută <Instrucțiuni> </pre> | sau | <pre> cât timp <condiție> execută <Instrucțiuni> sfârșit cât timp </pre> |
|---|-----|--|

- structură repetitivă cu test final

```

repetă
  <Instrucțiuni>
până când <condiție>

```

- structură repetitivă cu număr cunoscut de pași (unde valorile contorului includ limitele: <expInit>, respectiv <expFin>)

```

pentru <contor>←<expInit>,<expFin>[,<pas>] execută
  <Instrucțiuni>

```

sau

```

pentru <contor>←<expInit>,<expFin>[,<pas>] execută
  <Instrucțiuni>
sfârșit pentru

```

Subprogramele/modulele sunt definite în formatele:

```

subprogram <identificator> [(<parametri formali>)] [returnează <rezultate sau tipuri de date>]
  <Instrucțiuni>

```

sau

```

subprogram <identificator> [(<parametri formali>)] [returnează <rezultate sau tipuri de date>]
  <Instrucțiuni>
sfârșit subprogram

```

Cu apelul

```

Apel <identificator> [(<listă de parametri efectivi>)]

```

Un **element al unei liste indexate** se accesează precizând poziția ca indice (sau indici, separați prin virgulă), ai variabilei de tip listă: de exemplu a_i sau $a_{i,j}$.

Comentariile sunt precedate de două bare oblice // înainte și continuă până la sfârșitul liniei

Pentru operații aritmetice se utilizează simboluri standard ale operatorilor aritmetici: + (adunare), - (scădere), * (înmulțire), / (împărțire), ^ (ridicare la putere), √ (rădăcină pătrată), pentru partea întreagă a unui număr real se utilizează încadrarea expresiei între paranteze drepte, iar pentru restul împărțirii a două numere întregi se utilizează operatorul %.

Pentru exersarea și implementarea algoritmilor în limbaje de programare elevii pot utiliza **medii de dezvoltare (IDE)**, cum ar fi cele de mai jos.

pentru Python:

- PyCharm (variantele Community Edition și Educational Edition) - multiplatformă (Windows, Linux, Mac-OS etc.), plug-in (<https://www.jetbrains.com/pycharm/>)
- IDLE Python - multiplatformă (Windows, Linux, Mac-OS etc.);
- Spyder - multiplatformă (Windows, Linux, MacOS etc.), plug-in (<https://docs.spyder-ide.org/index.html>)
- Wing Wing (varianta Wing Personal) - multiplatformă (Windows, Linux, MacOS etc.);
- IDE Komodo - pentru dezvoltarea aplicațiilor web și mobile, multiplatformă (Windows, Linux, MacOS etc.)

pentru C++:

- Code Blocks - multiplatformă (Windows, Linux, MacOS etc.), plug-in (<https://www.codeblocks.org/>)

pentru Python și C++:

- Visual Studio Code - multiplatformă (Windows, Linux, MacOS etc.), plug-in (<https://vscode.dev>)

Instrumente online pentru implementarea soluțiilor

- Online GDB (<https://www.onlinegdb.com>)
- Programiz (<https://www.programiz.com/>)
- w3schools (<https://www.w3schools.com/>)
- Repl.it / Jupyter Notebooks (inclusiv pentru activități de programare colaborativă)

- MySQL Workbench (<https://dev.mysql.com/workbench/>)

Interpretoare Python

- CPython (www.python.org)
- PyPy (<https://www.pypy.org/download.html>)

Pentru **sprijinul activităților didactice de predare-învățare-evaluare** pot fi utilizate lecții interactive, tutoriale specifice, platforme de generare a testelor, ca cele recomandate mai jos.

Pentru generarea testelor:

- Kahoot! (<https://kahoot.com>), BookWidGets (<https://www.bookwidgets.com/>), Wayground (<https://wayground.com/>), Genially (<https://genially.com/>), WordWall (<https://wordwall.net>), Edcafe (<https://www.edcafe.ai/>)

Pentru obținerea unor mijloace de învățământ:

- Canva Education (<https://www.canva.com/education>) pentru creare de materiale vizuale, prezentări, postere și fișe de lucru; șabloane educaționale gratuite.
- MagicSchool (<https://www.magicschool.ai/>) platformă internă (sau locală) pentru crearea și distribuirea de resurse, lecții interactive și instrumente de evaluare.
- Livresq (<https://livresq.com/ro/>), bibliotecă de resurse educaționale interactive, platformă pentru crearea unor astfel de resurse;
- NEXTLAB.TECH (robo.nextlab.tech), un instrument modern de învățare prin practică, gamificare, IA și proiecte interactive.

Pentru învățare prin explorare și vizualizare, se recomandă utilizarea de diagrame și reprezentări grafice, simulatoare, instrumente interactive disponibile online, precum:

- HTML Maze (<https://www.htmlmaze.online/maze>) - instrument educațional pentru vizualizarea unor algoritmi, demonstrații pas cu pas etc.;
- Data Structure Visualizations (<https://www.cs.usfca.edu/~galles/visualization/Algorithms.html>) - o colecție interactivă de vizualizări pentru structuri de date și algoritmi
- Visualgo.net (<https://visualgo.net/>) - instrument educațional pentru vizualizarea algoritmilor, demonstrații pas cu pas pentru BFS/DFS, Dijkstra etc.;
- Graph Visualizer / Graph Online (<https://graphonline.ru/en/>) - instrument online de desenare și simulare a grafurilor orientate/neorientate, ponderate
- Algorithm Visualizer (<https://algorithm-visualizer.org/>) - platformă open-source pentru vizualizarea algoritmilor în mai multe limbaje (Python, JavaScript, C++), care conține animații pentru BFS și DFS, dar și pentru sortare, Backtracking, grafuri ponderate etc.
- Python Tutor (<https://pythontutor.com>) – permite rularea pas cu pas a codului Python, C++ cu vizualizarea memoriei și a fluxului de execuție
- CS Field Guide (<https://csfieldguide.org.nz/en/>) – resurse vizuale și jocuri interactive pentru concepte precum compresia datelor, criptare, rețele și algoritmi
- Draw.io (<https://app.diagrams.net>) - pentru a desena entități, attribute, relații și a conecta elementele prin linii care se pot salva local sau în Google Drive
- Lucidchart (<https://lucid.app/>) - platformă online dedicată creării de diagrame, care oferă modele predefinite pentru reprezentarea entităților și relațiilor
- Teachable Machine (<https://teachablemachine.withgoogle.com/>) sau Machine Learning for Kids (<https://machinelearningforkids.co.uk>) pentru a antrena modele simple (clasificare de imagini, recunoaștere de sunete), conectând activitățile la domenii diverse (biologie, arte, științe sociale)

GRUP DE LUCRU

| Nume și prenume | Grad didactic/Titlu științific | Instituție de apartenență, localitate, județ |
|---|---------------------------------------|--|
| CRĂCIUNESCU Georgeta Antonia Rodica | consilier | Ministerul Educației și Cercetării |
| ACIOBĂNIȚEI Iulian | conferențiar universitar, doctor | Academia Tehnică Militară „Ferdinand I”, București |
| ANTON Cristina Elena | profesor, gradul didactic I | Colegiul Național „Gh. M. Murgoci”, Brăila, județul Brăila |
| BOCA Alina Gabriela | profesor, gradul didactic I | Colegiul Național de Informatică „Tudor Vianu”, București |
| BORZA Diana-Laura | lector universitar, doctor | Universitatea Babeș Bolyai, Cluj-Napoca, județul Cluj |
| CÂRSTEA Claudia | conferențiar universitar, doctor | Academia Forțelor Aeriene „Henri Coandă”, Brașov, județul Brașov |
| CONTRAȘ Diana | profesor, gradul didactic I | Colegiul Național „Gheorghe Șincai”, Baia Mare, județul Maramureș |
| DIOSAN Laura-Silvia | profesor universitar, doctor | Universitatea Babeș-Bolyai, Cluj-Napoca, județul Cluj |
| DRAGOMIR-CONSTANTIN Florentina-Loredana | conferențiar universitar, doctor | Universitatea Națională de Apărare Carol I, București |
| DUMITRAN Adrian-Marius | lector universitar, doctor | Universitatea București, Facultatea de Matematică și Informatică |
| FLOREA Andrei | profesor, gradul didactic I | Colegiul Național „Ion Luca Caragiale”, București |
| IFTENE Adrian | profesor universitar, doctor | Universitatea „Alexandru Ioan Cuza”, Facultatea de Informatică, Iași, județul Iași |
| IORDAICHE Eugenia-Cristiana | profesor, gradul didactic I | Liceul Teoretic „Grigore Moisil”, Timișoara, județul Timiș |
| MARCU ALEXANDRA | profesor, gradul didactic I | Colegiul Național Militar „Tudor Vladimirescu”, Craiova, Dolj |
| MAIER Mariana-Ioana | lector universitar, doctor | Universitatea Babeș Bolyai, Cluj-Napoca, județul Cluj |
| MANZ Victor | profesor, gradul didactic I | Colegiul Național de Informatică „Tudor Vianu”, București |
| MARIUC Florin Constantin | profesor, gradul didactic I | Colegiul Național Militar „Ștefan cel Mare”, Câmpulung Moldovenesc, județul Suceava |
| MĂGUREANU Livia | cadru didactic asociat | Universitatea București, Facultatea de Matematică și Informatică |
| MODRIȘAN Adrian | profesor, gradul didactic I | Colegiul Național „Andrei Șaguna”, Brașov, județul Brașov |
| NAN Mihai | șef lucrări, doctor | Universitatea Națională de Științe și Tehnologie Politehnica București, Facultatea de Automatică și Calculatoare |
| NIȚU Alina | profesor, gradul didactic I | Liceul Teoretic „Ovidius”, Constanța, județul Constanța |

| Nume și prenume | Grad didactic/Titlu științific | Instituție de apartenență, localitate, județ |
|--------------------------|----------------------------------|--|
| NURLA Aidan | profesor, gradul didactic I | Colegiul Național Militar „Alexandru Ioan Cuza”, Constanța, județul Constanța |
| OANCEA Romana | conferențiar universitar, doctor | Academia Forțelor Terestre „Nicolae Bălcescu”, Sibiu, județul Sibiu |
| PETRESCU Manuela-Andreea | lector universitar, doctor | Universitatea Babeș-Bolyai, Cluj-Napoca, județul Cluj |
| PINTEA Adrian-Doru | profesor, gradul didactic I | Colegiul Național „Andrei Mureșanu”, Dej, județul Cluj |
| PINTEA Rodica | profesor, gradul didactic I | Liceul Teoretic „Radu Vlădescu”, Pătârlagele, județul Buzău |
| POP Florin | profesor universitar, doctor | Universitatea Națională de Științe și Tehnologie Politehnica București, Facultatea de Automatică și Calculatoare |
| POSTOLACHE Florin | lector universitar, doctor | Academia Navală „Mircea cel Bătrân” Constanța, Facultatea de Inginerie Marină, județul Constanța |
| SĂCUIU Silviu Eugen | profesor, gradul didactic I | Colegiul Național „Mihai Viteazul”, București |
| SPĂTARU Adrian | lector universitar, doctor | Universitatea de Vest, Timișoara, Facultatea de Matematică și Informatică, județul Timiș |
| STANCIU Diana | profesor, gradul didactic I | Colegiul Național Militar „Dimitrie Cantemir”, Breaza, județul Prahova |
| TEGLAȘ Maria | profesor, gradul didactic II | Colegiul Național Militar „Mihai Viteazul”, Alba Iulia, județul Alba |
| TÎMPLARU Roxana-Gabriela | profesor, gradul didactic I | Colegiul „Ștefan Obobleja”, Craiova, județul Dolj |
| UNGUREANU Florentina | profesor, gradul didactic I | Colegiul Național de Informatică, Piatra Neamț, județul Neamț |
| VINȚ Corina Elena | profesor, gradul didactic I | Colegiul Național de Informatică „Tudor Vianu”, București |

COORDONATORI/RESPONSABILI/CONSULTANȚI ȘTIINȚIFICI

| Nume și prenume | Grad didactic/Titlu științific | Instituție de apartenență, localitate, județ |
|--------------------|----------------------------------|--|
| PENEA Ștefania | consilier | Centrul Național pentru Curriculum și Evaluare |
| ȚOCA Livia Demetra | consilier | Centrul Național pentru Curriculum și Evaluare |
| ȚĂPUS Nicolae | profesor universitar, doctor | Academia Română |
| BORIGA Radu Eugen | conferențiar universitar, doctor | Universitatea București, Facultatea de Matematică și Informatică |