Programa școlară
pentru disciplina

Informatică

Clasa a XII-a

Curriculum de specialitate (CS)

Filiera teoretică, profilul real, specializarea matematică-informatică, clase cu predarea disciplinei informatică în regim intensiv

Învățământ liceal

- 2026 –

Anexa nr. 52
Programa școlară pentru disciplina *Informatică,* clasa a XII-a, curriculum de specialitate (CS)

- *Filiera teoretică, profilul real, specializarea matematică-informatică, clase cu predarea disciplinei informatică în regim intensiv*

NOTĂ DE PREZENTARE

Statutul disciplinei

Disciplina *informatică*, studiată în ciclul liceal, este o continuare a disciplinei *informatică și TIC*, studiată în gimnaziu, în trunchiul comun. În ciclul liceal, aceasta are rolul de a consolida și extinde domeniile de competență, aprofundând aspectele privind 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 studiază 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, clase cu predarea disciplinei *informatică* în regim intensiv, alocarea orară este:

- clasa a IX-a – 4 ore/săptămână, dintre care două ore pentru studiu teoretic și două ore pentru activități practice;
- clasa a X-a – 4 ore/săptămână, dintre care două ore pentru studiu teoretic și două ore pentru activități practice;
- clasa a XI-a – 7 ore/săptămână, dintre care patru ore pentru studiu teoretic și trei ore pentru activități practice;
- clasa a XII-a – 7 ore/săptămână, dintre care patru ore pentru studiu teoretic și trei ore pentru activități practice.

Activitățile practice sunt desfășurate obligatoriu în laboratorul de informatică, iar în baza Ordinului ministrului educației și cercetării nr. 6873/2025, pentru filiera teoretică, profilul real, specializarea matematică-informatică, clase cu predarea disciplinei *informatică* în regim intensiv, activitățile practice se desfășoară cu colectivul de elevi organizat pe grupe.

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 OME 6466/2024);
- Cadrul de competențe de inteligență artificială pentru elevi (AI competency framework for students), UNESCO 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 de liceu, 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 artă digitală.

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.

Studierea 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 (competențe generale, competențe specifice, 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 XII-a
CS 1.1. Identifică principalele caracteristici ale organizării datelor în cadrul unor modele conceptuale avansate - pentru proiectarea bazelor de date sau învățare automată, pentru a structura și accesa date în vederea prelucrării acestora
- *identificarea elementelor principale ale unui model entitate-relație (entitate, atribut, relație, identificator unic, tipuri de identificatori unici - natural/artificial, simplu/compus) în contextul gestionării informațiilor despre cărți, elevi și împrumuturile realizate într-o bibliotecă școlară* - *recunoașterea tipurilor de cardinalitate (1:1, 1:M, M:M) dintr-o listă de exemple de relații între entități pentru gestionarea unui magazin online* - *identificarea termenilor specifici (supertip, subtip, arc, transferabilitate), în contextul gestionării unui sistem de analiză a rețelelor de transport, prin evidențierea tipurilor de trasee* - *identificarea caracteristicilor unui subtip al unei entități (moștenire de atribute, relații comune) pentru un sistem de gestionare a angajaților, prin evidențierea atributelor moștenite* - *recunoașterea etapelor de preprocesare a datelor (curățare, normalizare, codificare) într-un flux de analiză a datelor provenite din senzori medicali* - *identificarea conceptelor de bază (set de antrenare, set de testare, caracteristici, etichetă) într-un proiect practic ce presupune clasificarea automată a e-mailurilor în unele de tip spam/non-spam*
CS 1.2. Identifică specificul, caracteristicile și etapele unor strategii de normalizare a modelului conceptual al unei probleme de gestiune
- *enumerarea caracteristicilor datelor care corespund primei forme normale (FN1), în contextul unei baze de date pentru o bibliotecă, unde unele cărți au mai mulți autori* - *enumerarea caracteristicilor datelor care corespund celei de a doua forme normale (FN2), în contextul unei baze de date pentru o bancă, unde unele tabele au identificator compus* - *enumerarea caracteristicilor datelor care corespund celei de a treia forme normale (FN3), în contextul unei baze de date pentru o școală, unde pentru fiecare elev sunt memorate datele părinților*
CS 1.3. Identifică specificul, caracteristicile și etapele unor algoritmi specializați pe clase de probleme, pentru a îi putea utiliza în învățarea automată
- *enumerarea caracteristicilor învățării supervizate în contextul aplicării pentru soluționarea unor probleme practice pentru care avem acces la date etichetate de experți* - *enumerarea caracteristicilor învățării nesupervizate în contextul aplicării pentru soluționarea unor probleme practice pentru care nu avem acces la etichete* - *recunoașterea termenilor de bază (de exemplu, învățare supervizată, învățare nesupervizată, clasificare, regresie, clusterizare) în contextul unor situații practice din viața reală* - *identificarea caracteristicilor algoritmului K-means într-un context de clasificare a clienților după comportamentul de cumpărare* - *enumerarea pașilor principali ai unui algoritm de învățare bazat pe ajustarea ponderilor (feedforward și backpropagation)* - *reamintirea situațiilor în care rețelele neuronale pot fi aplicate (recunoaștere imagini, voce, text), prin exemple simple din viața cotidiană*
CS 1.4. Identifică principalele comenzi SQL și ale limbajului de programare pentru a identifica soluția adecvată prelucrării datelor organizate în baze de date, în vederea rezolvării eficiente a problemelor informatice de gestiune
- *identificarea tipurilor de comenzi SQL pentru gestionarea unei baze de date a unei biblioteci, prin clasificarea fiecărei comenzi în categoria corespunzătoare* - *descrierea structurii de bază a unei comenzi SELECT pentru interogarea unei baze de date a unui magazin online, prin enumerarea componentelor acesteia* - *recunoașterea comenzilor SQL pentru gestionarea bazei de date a unei clinici medicale, prin acțiuni corespunzătoare asupra tabelelor* - *enumerarea claselor și metodelor principale ale limbajului Python utilizate pentru conectarea la o bază de date pentru o aplicație de evidență a produselor* - *identificarea părților componente ale unui cod simplu de conectare la o bază de date (nume al clasei, metodă adecvată, parametri) pentru un script Python care accesează baza de date a unei clinici medicale* - *recunoașterea rolului bibliotecilor uzuale (sqlite3, mysql.connector) într-un fragment de cod Python pentru două aplicații diferite*

Clasa a XII-a
CS 1.5. Identifică principalele elemente ale limbajului de programare utilizate pentru prelucrarea datelor organizate în învățarea automată, în vederea rezolvării eficiente a problemelor informatice
- *enumerarea bibliotecilor Python utilizate în învățarea automată (Pandas, Matplotlib, Scikit-learn) și precizarea scopului fiecăreia pentru înțelegerea ecosistemului de lucru cu date, specificând că Pandas gestionează date tabulare, Matplotlib vizualizări grafice și Scikit-learn algoritmi de machine learning* - *identificarea claselor principale (DataFrame, Series, LinearRegression) în fragmente de cod Python pentru recunoașterea structurilor de date într-un script de analiză a vânzărilor, distingând DataFrame ca tabel complet, Series ca o coloană și LinearRegression ca model de regresie liniară* - *recunoașterea tipurilor de grafice (linie, bare, histogramă, box-plot, scatter) generate de Matplotlib pentru interpretarea corectă a vizualizărilor dintr-un raport de analiză, identificând graficul de linie pentru evoluții temporale, bare pentru comparații categoriale, histogramă pentru distribuții, box-plot pentru statistici descriptive și scatter pentru corelații* - *recunoașterea modului de creare a array-urilor prin identificarea metodelor de bază din NumPy pentru operații cu liste de valori (np.array, np.zeros, np.ones, np.arange)* - *identificarea operațiilor uzuale de algebră liniară disponibile în NumPy, prin etichetarea fiecărei operații (înmulțire matriceală, transpunere, determinant, inversare)*

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 XII-a
CS 2.1. Explică principiile de organizare a datelor în cadrul unor modele conceptuale avansate - pentru proiectarea bazelor de date sau învățare automată, pentru a le gestiona eficient
- *explicarea diferenței dintre o entitate și o instanță a unei entități prin exemplul gestionării informațiilor despre elevi, profesori și discipline într-o școală* - *explicarea modului în care o relație M:M se transformă în două relații 1:M, de regulă barate, introducând o entitate de intersecție, prin exemplul unui sistem de notare a elevilor la fiecare disciplină, de-a lungul unui an de studiu* - *interpretarea unei diagrame simple entitate-relație și explicarea sensului simbolurilor utilizate (bare, opționalități, cardinalități) pentru un sistem de gestionare a comenzilor într-un restaurant, unde se analizează relațiile dintre clienți, comenzi și produse* - *explicarea diferenței dintre „supertip” și „subtip” în cadrul unui model conceptual pentru un sistem de gestionare a angajaților, prin evidențierea tipurilor și subtipurilor în cazul angajaților care lucrează part-time, full-time sau sezonieri* - *interpretarea unei diagrame entitate-relație care conține arce și formularea unei explicații a rolului acestora (relații alternative mutual exclusive) pentru un sistem de gestionare a transportului urban, evidențiind tipul de transport care poate fi ales la un moment dat (troleu, autobuz, metrou)* - *explicarea diferenței dintre date brute și date preprocesate folosind ca exemplu un set de date despre pacienți (unde unele valori lipsesc sau sunt exprimate în unități de măsură diferite)* - *ilustrarea prin exemple a modului în care structura datelor influențează performanța modelului într-un caz practic de predicție a prețului apartamentelor* - *explicarea, pe baza unui exemplu din domeniul sănătății, a modului în care datele incomplete sau neomogene pot afecta performanța unui model de diagnostic automat*
CS 2.2. Explică etapele unor strategii de normalizare a modelului conceptual al unei probleme de gestiune
- *explicarea importanței normalizării unei baze de date pentru un sistem de gestionare a comenzilor unui magazin online, prin evidențierea reducerii redundanțelor, a prevenirii inconsistențelor și a simplificării prelucrării datelor* - *explicarea diferențelor dintre prima, a doua și a treia formă normală, pentru un sistem de gestionare a comenzilor unui magazin online* - *interpretarea unui model conceptual dat pentru un sistem de gestionare a vânzărilor unui magazin, prin evidențierea respectării formelor normale*
CS 2.3. Explică etapele unor algoritmi specializați pe clase de probleme, pentru a îi putea utiliza în învățarea automată
- *explicarea principiilor învățării nesupervizate și clusterizării pe baza unui exemplu de grupare a utilizatorilor unei platforme video în funcție de preferințe* - *explicarea diferențelor dintre clasificare și regresie pe baza unui set de date cu vânzări lunare (regresie) versus tipul clientului (clasificare)* - *ilustrarea modului în care algoritmul KNN utilizează vecinii apropiați pentru a clasifica un element într-un scenariu simplu de clasificare a florilor (setul Iris)* - *explicarea modului în care alegerea atributelor poate afecta performanțele unui algoritm de învățare automată utilizat pentru rezolvarea unei probleme de predicție a prețului unui apartament* - *explicarea modului de funcționare a unui neuron artificial, prin reprezentarea grafică a intrărilor, ponderilor și funcției de activare* - *explicarea modului în care o rețea perceptron procesează informația și își ajustează conexiunile pentru a învăța din erori*

Clasa a XII-a
CS 2.4. Explică rolul componentelor principalelor comenzi SQL și ale limbajului de programare pentru a identifica soluția adecvată prelucrării datelor organizate în baze de date, în vederea rezolvării eficiente a problemelor informatice de gestiune
- *explicarea diferenței dintre comenzile DDL și DML pentru gestionarea bazei de date a unei școli, prin evidențierea acțiunilor corespunzătoare asupra structurii, respectiv asupra conținutului tabelelor* - *explicarea utilizării funcțiilor agregate într-o interogare pentru analiza vânzărilor dintr-un supermarket, prin rezultatul obținut în urma aplicării lor* - *interpretarea unui exemplu de interogare SQL și explicarea rezultatului pe baza datelor afișate pentru o bază de date cu angajați și departamente, prin analizarea pas cu pas a fiecărei clauze* - *explicarea rolului unei vederi (view) și a avantajelor utilizării acesteia pentru simplificarea accesului la date* - *interpretarea unei secvențe SQL care folosește comenzi precum COMMIT și ROLLBACK, în contextul unei tranzacții bancare de transfer, prin explicarea efectului fiecărei comenzi asupra bazei de date* - *explicarea, în propriile cuvinte, a diferenței dintre obiectul de tip Connection și cel de tip Cursor pentru înțelegerea arhitecturii de lucru cu baze de date în Python, evidențiind faptul că obiectul de tip Connection reprezintă legătura fizică la baza de date (sesiunea de comunicare), iar obiectul de tip Cursor este cel care execută comenzile SQL și gestionează rezultatele, ilustrând analogia cu o carte (Connection) și un semn de carte (Cursor) care marchează poziția curentă* - *descrierea pașilor necesari pentru realizarea unei conexiuni la o bază de date și executarea unei interogări pentru un script Python care accesează o bază de date SQLite cu informații despre produse, detaliind secvența: importarea bibliotecii (sqlite3), crearea conexiunii cu connect(), crearea obiectului Cursor cu cursor(), executarea interogării cu execute(), preluarea rezultatelor cu fetchall() sau fetchone(), și închiderea resurselor cu close()* - *interpretarea unui fragment de cod Python care utilizează obiecte de tip Connection și Cursor, prin explicarea secvenței de operații efectuate pentru a extrage date din tabela Comenzi, identificând fiecare linie de cod, rolul acesteia (conectare, executare, preluare date, închidere), ordinea de executare și modul în care datele sunt transferate de la baza de date în vederea prelucrării lor*
CS 2.5. Explică rolul principalelor elemente ale limbajului de programare utilizate pentru prelucrarea datelor în învățarea automată, în vederea rezolvării eficiente a problemelor informatice
- *explicarea modului în care un obiect DataFrame reprezintă datele sub formă de tabel pentru înțelegerea structurii de date Pandas, prin organizarea informațiilor* - *explicarea modului în care array-urile NumPy și DataFrame-urile Pandas stochează și manipulează datele pentru alegerea structurii potrivite în analiza datelor* - *interpretarea unui fragment de cod care construiește o regresie liniară simplă cu Scikit-learn pentru predicția prețurilor unor apartamente pe baza suprafeței, prin antrenarea unui model pe datele disponibile* - *explicarea diferențelor dintre operațiile cu liste Python și array-uri NumPy, prin descrierea avantajelor utilizării NumPy în învățare automată* - *explicarea rezultatelor operațiilor care implică broadcasting în NumPy, pentru înțelegerea adunării unui scalar la o matrice sau înmulțirea unui vector linie cu unul coloană*

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

Clasa a XII-a
CS 3.1. Utilizează modul de organizare și prelucrare a datelor în cadrul unor modele conceptuale avansate – la proiectarea bazelor de date sau la învățare automată, pentru a rezolva probleme de gestionare a datelor
- *aplicarea regulilor pentru transformarea unei relații M:M prin exemplul unui sistem de închiriere filme, unde se utilizează regula introducerii unei entități de intersecție, pentru împrumuturi* - *completarea unei diagrame entitate-relație parțiale cu marcatorii adecvați pentru atribute, respectând regulile de identificare unică pentru un sistem de gestiune a vânzărilor, care conține entitățile PRODUS, CLIENT și COMANDĂ* - *completarea unei diagrame entitate-relație parțiale cu supertipuri și subtipuri corespunzătoare unui scenariu dat, prin exemplul mijloacelor de transport pentru persoane* - *aplicarea regulilor de reprezentare pentru plasarea unui arc pe trei relații posibile, respectând regula de exclusivitate prin exemplul unui sistem de gestionare a rezervărilor la un hotel* - *aplicarea unor operații de augmentare a datelor (de exemplu, rotirea sau scalarea imaginilor) pentru un set de imagini cu fructe* - *aplicarea operației de normalizare a valorilor numerice ale unui set de date cu puține exemple provenit dintr-o bază de date cu indicatori economici, manual, prin calculul valorilor normalizate pe baza valorilor minime și maxime, pentru a aduce toate valorile pe aceeași scară* - *aplicarea unor criterii pentru separarea unui set de date despre rezultate școlare în seturi de antrenare și testare, în vederea asigurării existenței unor date noi care să fie utilizate pentru evaluarea unui algoritm de învățare automată*

Clasa a XII-a
CS 3.2. Aplică strategii de normalizare a modelului conceptual al unei probleme de gestiune
- *aplicarea strategiei pentru obținerea primei forme normale (FN1) pentru un sistem de gestionare a comenzilor unui magazin online, prin tratarea adecvată a fiecărui atribut al unei entități care nu are un caracter atomic, fiind format din mai multe componente care sunt utilizate independent* - *aplicarea strategiei pentru obținerea celei de a doua forme normale (FN2) pentru un model conceptual al unei baze de date, prin tratarea adecvată a dependențelor unor atribute de componente ale identificatorului unic compus, pentru un sistem de gestiune a activităților nonformale dintr-o școală* - *aplicarea strategiei pentru obținerea celei de a treia forme normale (FN3) a unui model conceptual al unei baze de date, prin tratarea adecvată a dependențelor unor atribute de alte atribute diferite de identificatorul unic, cu respectarea integrității datelor pentru un sistem de gestiune a angajaților și departamentelor unei companii*
CS 3.3. Aplică algoritmi specializați pe clase de probleme, pentru a îi putea utiliza în învățarea automată
- *aplicarea algoritmului K-means pe un set de date cu coordonate 2D reprezentând sediile a 15 magazine dintr-un oraș în vederea identificării a trei zone optime pentru amplasarea depozitelor de distribuție, urmărind cum centroizii se deplasează progresiv către pozițiile finale și cum magazinele sunt reatribuite la fiecare iterație* - *aplicarea algoritmului de regresie liniară pentru soluționarea unei probleme de predicție a prețurilor caselor pornind de la un set de date ce conține cinci proprietăți pentru care cunoaștem suprafața și prețul* - *aplicarea algoritmului de clasificare KNN pentru un set de date simplu cu fructe (mere, portocale, banane) descrise prin greutate și diametru* - *simularea funcționării unui neuron artificial, utilizând valori numerice simple pentru a calcula ieșirea pe baza intrărilor și ponderilor* - *aplicarea principiilor rețelelor perceptron pentru rezolvarea unei probleme simple de clasificare (recunoașterea formelor geometrice)*
CS 3.4. Utilizează principalele comenzi SQL și ale limbajului de programare pentru a identifica soluția adecvată prelucrării datelor organizate în baze de date, în vederea rezolvării eficiente a problemelor informatice de gestiune
- *aplicarea unei interogări SQL pentru baza de date a unui liceu, pentru afișarea elevilor cu media peste 9, în ordine alfabetică* - *aplicarea unei comenzi JOIN pentru obținerea unui raport școlar, prin extragerea informațiilor din două tabele relaționate* - *aplicarea unei funcții de conversie într-o interogare pentru a transforma un tip de date numeric în șir de caractere prin generarea de etichete cu prețuri într-un magazin* - *aplicarea comenzilor GRANT și REVOKE pentru gestionarea drepturilor de acces ale diferiților utilizatori, într-un sistem de rezervări hoteliere* - *utilizarea comenzilor SAVEPOINT, ROLLBACK și COMMIT, pentru a realiza tranzacții cu revenire la un punct stabilit, pentru procesarea comenzilor unui magazin online* - *utilizarea instrumentelor limbajului Python pentru conectarea la o bază de date SQLite și afișarea conținutului unei tabele pentru vizualizarea produselor dintr-un magazin, prin crearea conexiunii, a cursorului, executarea interogării și afișarea rezultatelor* - *utilizarea metodelor execute() și fetchall() și a comenzii SELECT pentru preluarea și afișarea rezultatelor unei interogări în vederea extragerii tuturor comenzilor dintr-o perioadă specificată* - *utilizarea metodelor execute(), close() și commit() și a comenzii INSERT INTO, pentru a insera date într-o tabelă, folosind conexiunea și cursorul corespunzător și închizând conexiunea corect, în contextul adăugării unui client nou în tabela Clienti*
CS 3.5. Utilizează principalele elemente ale limbajului de programare pentru prelucrarea datelor în învățarea automată, în vederea rezolvării eficiente a problemelor informatice
- *utilizarea metodelor din Pandas pentru citirea unui set de date dintr-un fișier Note.csv și afișarea valorilor statistice de bază (medie, minim, maxim, deviație standard) în contextul analizei situației școlare* - *aplicarea metodelor din Matplotlib pentru reprezentarea grafică a datelor privind vânzările lunare ale unui magazin, prin realizarea unor grafice adecvate (de exemplu, grafic liniar, diagramă cu bare)* - *utilizarea metodelor fit() și predict() ale clasei LinearRegression din Scikit-learn pentru a antrena un model de regresie liniară și realizarea unei predicții pentru consumul de combustibil pe baza greutății mașinii și a distanței parcurse* - *aplicarea funcțiilor și operațiilor din biblioteca NumPy pentru rezolvarea unui sistem de ecuații liniare, în vederea determinării prețurilor unor produse pe baza unui model de cost* - *aplicarea operațiilor numerice din biblioteca NumPy pentru implementarea explicită a normalizării (Min–Max) și standardizării (Z-score) unui set de date, prin calculul minimului, maximului, mediei și deviației standard, în vederea construirii unei matrice de caracteristici pentru antrenarea unui model de învățare automată*

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 XII-a
CS 4.1. Analizează caracteristicile, modul de prelucrare, avantajele și dezavantajele utilizării unor modele conceptuale avansate – pentru proiectarea bazelor de date sau pentru învățare automată, pentru a alege structura adecvată în rezolvarea unor probleme concrete

Clasa a XII-a
- *analizarea unui model conceptual dat, corespunzător activității de programare a cursanților dintr-o școală de conducători auto, pentru identificarea corespondenței dintre scenariu și entitățile și relațiile modelului* - *analizarea unui model entitate-relație existent pentru a identifica erorile de cardinalitate pentru un sistem de gestionare a rezervărilor la un hotel, prin examinarea relațiilor dintre clienți, camere și rezervări* - *analizarea unui model entitate-relație dat pentru a identifica erorile de redundanță pentru un sistem de gestionare a comenzilor într-un magazin online* - *analizarea unei diagrame entitate-relație pentru a detecta și corecta relații redundante sau incorect modelate pentru un sistem de gestionare a angajaților, cu subtipuri care omit atributele moștenite sau relațiile comune* - *compararea a două variante de model conceptual pentru același scenariu prin identificarea diferențelor în utilizarea arcelor sau a transferabilității pentru un sistem de gestionare a proiectelor* - *analizarea structurii unui set de date real (număr de atribute, tipuri de variabile, valori lipsă), pentru identificarea pașilor de preprocesare necesari* - *compararea diferitelor metode de transformare a datelor (scalare min-max vs. standardizare), pentru identificarea contextului potrivit pentru fiecare, pornind de la o problemă din viața reală* - *analizarea corelațiilor dintre variabile într-un set de date despre obiceiuri alimentare și starea de sănătate, pentru a identifica atributele relevante pentru predicția finală*
CS 4.2. Analizează aplicabilitatea, avantajele și dezavantajele utilizării unor strategii de normalizare a modelului conceptual al unei probleme de gestiune
- *analizează avantajele unui model conceptual ce respectă prima formă normală (FN1), comparativ cu un model care nu respectă această formă, în contextul obținerii orașului de domiciliu al unei persoane, pentru cazul în care acesta este inclus în adresă (utilizarea unei părți a unui atribut) și pentru cazul în care el reprezintă un câmp individual (utilizarea acestuia ca o valoare atomică)* - *detectarea dependențelor funcționale dintre atribute și clasificarea lor pentru un sistem de gestiune a angajaților și proiectelor dintr-o companie, prin identificarea dependențelor parțiale și a celor tranzitive* - *compararea a două modele conceptuale ale aceleiași situații prin evidențierea celei care respectă principiile normalizării pentru un sistem de gestionare a comenzilor și clienților unui magazin de electronice*
CS 4.3. Analizează comportamentul, aplicabilitatea, avantajele și dezavantajele utilizării unor algoritmi specializați pe clase de probleme pentru învățarea automată
- *analizarea diferențelor dintre învățarea supervizată și cea nesupervizată, prin completarea unui tabel comparativ cu exemple concrete (de exemplu, predicția notelor vs. gruparea elevilor după interese)* - *compararea performanței algoritmilor K-means, regresie liniară și KNN, prin aplicarea fiecăruia pe seturi de date adecvate problemelor pe care le pot rezolva, din perspectiva metricii specifice fiecărui tip de problemă (de exemplu, coeficientul Silhouette pentru K-means, eroarea medie pătratică pentru regresie liniară, acuratețea sau matricea de confuzie pentru KNN)* - *interpretarea grafică a rezultatelor obținute de diferiți algoritmi, identificând modelele de date și abaterile față de tendință* - *analizarea unui caz (de exemplu, recomandarea de filme) pentru a identifica tipul de algoritm de învățare automată cel mai potrivit și justificarea alegerii în funcție de natura problemei și a datelor disponibile* - *analizarea structurii unei rețele neuronale, prin identificarea rolului fiecărui strat (intrare, ascuns, ieșire) într-o schemă grafică* - *compararea performanței unei rețele perceptron cu o rețea cu mai multe straturi, pe același set simplu de date*
CS 4.4. Analizează aplicabilitatea, avantajele și dezavantajele utilizării comenzilor SQL și a elementelor specifice limbajului de programare, în vederea rezolvării eficiente a problemelor informatice de gestiune
- *examinarea unei comenzi SQL pentru o interogare care extrage comenzile dintr-un sistem de gestiune a unui magazin online, prin identificarea eventualelor erori de sintaxă sau de logică* - *compararea a două interogări care obțin același rezultat, dar folosesc metode diferite (JOIN vs. subinterogare) pentru afișarea produselor comandate de un client specific dintr-un sistem e-commerce, prin evaluarea avantajelor și dezavantajelor fiecărei abordări* - *examinarea părților componente ale unei interogări complexe pentru un raport de vânzări lunar, prin explicarea rolului fiecăreia* - *examinarea corectitudinii unui set de comenzi SQL pentru actualizarea stocului după vânzări, prin identificarea punctelor vulnerabile* - *compararea a două metode de control al accesului (prin GRANT și prin utilizatori dedicați) pentru un sistem de salarizare, prin identificarea avantajelor fiecărei metode* - *analizarea unui cod Python care realizează o conexiune la o bază de date și identificarea posibilelor erori (lipsa commit(), conexiune neînchisă) pentru un script care actualizează prețurile produselor, verificând dacă lipsește commit() după UPDATE (modificările nu se salvează), dacă conexiunea rămâne deschisă (resurse blocate) sau dacă nu există tratare de excepții* - *compararea a două modalități de executare în Python a comenzilor SQL (prin execute() și prin executemany()) și determinarea cazului potrivit fiecăreia pentru inserarea de date în tabela Elevi, stabilind că execute() este eficient pentru o*

Clasa a XII-a
singură înregistrare, iar executemany() este optim pentru inserarea mai multor înregistrări dintr-o listă, reducând numărul de apeluri - *descompunerea unui exemplu complet (creare, inserare, interogare, închidere) pentru a evidenția ordinea corectă a apelurilor de metode pentru un script (de exemplu, pentru gestiunea unei biblioteci), secvențializând: connect() → cursor() → execute(CREATE TABLE) → execute(INSERT) → commit() → execute(SELECT) → fetchall() → close(), identificând dependențele dintre pași*
CS 4.5. Analizează aplicabilitatea, avantajele și dezavantajele utilizării unor elemente ale limbajului de programare, pentru a identifica soluția adecvată prelucrării datelor în învățarea automată, în vederea rezolvării eficiente a problemelor informatice
- *analizarea unui program Python care citește, curăță și vizualizează datele pentru a identifica ordinea corectă a pașilor de prelucrare pentru un script ce analizează tranzacțiile comerciale* - *compararea rezultatelor obținute cu funcția describe() din Pandas cu valorile calculate manual pe baza operațiilor de bază cu liste pentru statisticile de bază (medie, minim, maxim, deviație standard) pe un set de date cu temperaturi, verificând concordanța calculelor* - *analizarea unui grafic complex pentru a explica modul în care datele sunt combinate și afișate în Matplotlib pentru un panou de control cu grafice multiple despre performanța elevilor* - *compararea diferitelor metode NumPy pentru aceeași operație, prin investigarea diferențelor de comportament și performanță pentru înmulțirea matricilor în diverse scenarii* - *analizarea structurii unui array multidimensional, prin determinarea axelor și dimensiunilor pentru operații de agregare*

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

Clasa a XII-a
CS 5.1. Argumentează alegerile făcute în cadrul proiectării unor modele conceptuale avansate - pentru proiectarea bazelor de date sau pentru învățare automată, în vederea rezolvării unor probleme informatice concrete
- *evaluarea unui model entitate-relație din perspectiva respectării regulilor de identificare unică a instanțelor pentru un sistem de gestionare a închirierilor de mașini* - *evaluarea unui model entitate-relație din punctul de vedere al rezolvării adecvate a relațiilor M:M, pentru un sistem de gestionare a unei biblioteci digitale* - *justificarea deciziei de a folosi o relație ierarhică în locul uneia recursive, pentru un sistem de gestionare a structurii organizaționale a unei companii, prin analiza relației dintre angajați și manageri* - *evaluarea corectitudinii utilizării arcelor într-un model entitate-relație dat, pentru un sistem de gestionare a transportului urban* - *justificarea alegerii unui anumit mod de reprezentare a datelor sub formă de subtipuri sau entități separate, pentru un sistem de gestionare a angajaților* - *evaluarea calității unui set de date privind tranzacțiile bancare (din punctul de vedere al completitudinii și relevanței) înainte de utilizarea într-un algoritm de învățare automată* - *evaluarea impactului unei variante de normalizare asupra preciziei unui model de învățare automată aplicat pe un set de date despre diabet* - *argumentarea deciziei de a folosi codificarea de tip one-hot în locul asocierii unei etichete pentru variabile categorice ce descriu tipul unei mașini (sedan, SUV), în funcție de algoritmul de învățare automată utilizat*
CS 5.2. Evaluează corectitudinea și eficiența modelelor conceptuale ale unei probleme de gestiune din punctul de vedere al normalizării
- *evaluarea corectitudinii procesului de normalizare în cadrul bazelor de date aplicat pentru un sistem de gestiune al comenzilor și clienților unui magazin* - *argumentarea avantajelor și dezavantajelor utilizării formelor normale pentru un sistem de gestionare a comenzilor și stocurilor unui depozit, prin evidențierea avantajelor precum reducerea redundanțelor și creșterea consistenței datelor și a dezavantajelor precum creșterea numărului de tabele și complexitatea interogărilor* - *justificarea păstrării sau eliminării unei relații dintr-un model ERD în funcție de impactul asupra integrității și redundanței datelor pentru un sistem de gestionare a comenzilor și produselor unui magazin* - *identificarea atributelor care provoacă denormalizarea într-o bază de date pentru gestionarea vânzărilor dintr-un magazin, prin evidențierea atributelor care se repetă sau depind de alte atribute decât cheia primară*
CS 5.3. Evaluează corectitudinea și eficiența soluțiilor cu algoritmi specializați pe clase de probleme pentru învățare automată
- *evaluarea performanței unui model de regresie liniară pentru estimarea consumului de combustibil pe baza greutății mașinii, prin compararea valorilor reale cu cele prezise și calcularea erorii medii (Mean Absolute Error sau Mean Squared Error)* - *argumentarea alegerii algoritmului potrivit (K-means, regresie liniară sau KNN) în funcție de scopul analizei și tipul datelor*

Clasa a XII-a
- *formularea concluziilor privind precizia și utilitatea modelului KNN pentru clasificarea tipurilor de fructe pe baza caracteristicilor lor (greutate, culoare, diametru), folosind o valoare concretă a lui k (de exemplu, k=3), prin compararea valorilor reale și prezise obținute* - *argumentarea alegerii tipului de rețea sau a algoritmului de învățare potrivit (de exemplu, perceptron) pentru o anumită problemă practică* - *aprecierea limitelor modelelor de rețele neuronale, prin identificarea situațiilor în care acestea pot produce erori sau rezultate imprevizibile*
CS 5.4. Evaluează corectitudinea și eficiența secvențelor de cod sau aplicațiilor care utilizează principalele comenzi SQL și ale limbajului de programare pentru prelucrarea datelor organizate în baze de date, în vederea rezolvării eficiente a problemelor informatice de gestiune
- *evaluarea eficienței unei interogări SQL pentru un raport al unui sistem bancar care afișează clienții cu tranzacții suspecte, prin analizarea performanței interogării actuale și propunerea de eventuale versiuni îmbunătățite* - *validarea respectării principiilor de integritate referențială pentru baza de date a unei universități, prin examinarea consecințelor ștergerii unor date asupra înregistrărilor dependente* - *justificarea alegerii unui anumit tip de funcție (agregată vs. scalară) pentru obținerea rezultatului dorit în contexte diferite dintr-un sistem de resurse umane prin calcularea salariului mediu pe departament versus formatarea numelui complet al angajatului* - *argumentarea alegerii între o vedere și o interogare directă, în funcție de cerințele de securitate și performanță pentru afișarea datelor* - *justificarea folosirii comenzilor COMMIT sau ROLLBACK pentru un scenariu de lucru colaborativ prin actualizări simultane într-un sistem de rezervări* - *evaluarea eficienței unui program Python care lucrează cu baze de date, din punctul de vedere al timpului de executare și al gestionării resurselor pentru o aplicație de import masiv de tranzacții, analizând dacă executemany() este folosit corect, conexiunea se deschide/închide repetat ineficient și dacă fetchall() consumă prea multă memorie pentru seturi mari de date* - *argumentarea alegerii unei metode de conectare (sqlite3 local vs. mysql.connector extern) în funcție de scopul aplicației pentru proiectarea unui sistem de evidență, justificând sqlite3 pentru aplicații desktop single-user cu portabilitate (de exemplu, catalog personal) sau mysql.connector pentru aplicații multi-user cu acces concurent (de exemplu, sistem școlar)* - *justificarea utilizării blocurilor try-except-finally pentru gestionarea erorilor și închiderea sigură a conexiunilor pentru un modul de actualizare a datelor critice, demonstrând că try asigură execuția codului riscant, except prinde erori de conexiune/sintaxă SQL fără a opri programul, iar finally garantează close() chiar dacă apar excepții*
CS 5.5. Evaluează corectitudinea și eficiența aplicațiilor care utilizează elemente specifice de limbaj de programare pentru prelucrarea datelor în învățarea automată, în vederea rezolvării eficiente a problemelor informatice
- *evaluarea performanței unui model de regresie liniară pentru predicția prețurilor imobiliare, prin compararea valorilor reale și prezise și calcularea erorii medii pătratice (Mean Squared Error) folosind mean_squared_error din Scikit-learn, pentru setul de date de test* - *argumentarea alegerii tipului de reprezentare grafică (de exemplu, diagramă de dispersie vs. histogramă) în funcție de tipul de date pentru vizualizarea relației notă-timp de studiu, pentru un raport privind notele și timpul de studiu dintr-o platformă de cursuri online* - *justificarea utilizării unei metode de curățare a datelor într-un set concret privind numărul de exerciții rezolvate de elevi pentru fiecare temă, cu valori incomplete corespunzătoare zilelor în care elevii au lipsit, prin evaluarea impactului asupra rezultatelor* - *argumentarea alegerii NumPy față de alte biblioteci pentru operații de algebră liniară în contextul învățării automate, prin justificarea criteriilor de performanță* - *evaluarea limitărilor NumPy în procesarea seturilor mari de date, prin observarea situațiilor în care operațiile devin lente sau consumul de memorie crește semnificativ și identificarea nevoii de divizare a datelor*

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

Clasa a XII-a
CS 6.1. Proiectează reprezentări personalizate ale unor modele conceptuale avansate - pentru baze de date sau adecvate pentru învățare automată, în vederea rezolvării unor probleme
- *proiectarea unui model entitate-relație personalizat pentru un sistem de gestionare a rezervărilor la un hotel, prin determinarea entităților și a relațiilor dintre acestea* - *proiectarea unui model entitate-relație personalizat pentru un sistem de gestionare a istoricului comenzilor și prețurilor într-un magazin online, pentru a asigura funcționarea corectă a bazei de date* - *îmbunătățirea unui model entitate-relație existent, cu justificarea modificărilor aduse pentru un sistem de gestionare a rezervărilor la un restaurant* - *proiectarea unei diagrame entitate-relație care include tipuri și subtipuri pentru un sistem de gestionare a unei școli* - *crearea unei versiuni alternative a unui model entitate-relație existent, prin optimizarea cu arce, pentru gestionarea traficului*

Clasa a XII-a
- *schematizarea fluxului de prelucrare a datelor într-un proiect de învățare automată (colectare, curățare, transformare, antrenare) utilizat pentru detecția automată a unor probleme ale plămânilor, pornind de la imagini medicale de tip CT* - *proiectarea unui model conceptual al unui set de date (atribute, relații, etichete) pentru o problemă de predicție (de exemplu, performanța elevilor, vânzări, temperaturi)* - *crearea unui set de date sintetic care simulează valorile unor senzori de mediu (temperatură, umiditate, presiune) și care poate fi folosit pentru antrenarea unui model de predicție*
CS 6.2. Proiectează soluții cu aplicarea personalizată a strategiilor pentru normalizarea modelului conceptual al unei probleme de gestiune
- *proiectarea unui modelul conceptual pentru o aplicație de gestiune prin respectarea celor trei forme normale, pentru un magazin de electronice* - *elaborarea unui model conceptual normalizat pentru un sistem de gestionare a închirierilor de mașini, prin aplicarea fiecărei etape de normalizare* - *crearea unei reprezentări grafice (ERD) care demonstrează trecerea progresivă prin 1NF, 2NF și 3NF pentru un sistem de gestionare a rezervărilor la un hotel, prin prezentarea ERD inițial, apoi ERD după aplicarea fiecărei forme de normalizare*
CS 6.3. Proiectează soluții cu aplicarea personalizată a algoritmilor specializați pe clase de probleme pentru învățare automată
- *proiectarea unui mini-experiment de învățare automată, pornind de la un set de date colectat de elevi (de exemplu, obiceiuri de lectură, preferințe muzicale)* - *crearea unui model simplu de clasificare sau regresie utilizând aplicații educaționale (Orange Data Mining, Teachable Machine, Excel)* - *elaborarea unui scurt raport digital care descrie etapele aplicării algoritmului ales (pregătirea datelor, antrenarea modelului, interpretarea rezultatelor)* - *proiectarea unui experiment pentru testarea diferitelor rate de învățare în antrenarea perceptronului, prin combinarea rezultatelor într-un raport vizual pentru problema clasificării pacienților cu anemie pe baza parametrilor sangvini*
CS 6.4. Implementează secvențe de cod sau aplicații care integrează în mod personalizat comenzi SQL și elemente ale limbajului de programare pentru prelucrarea datelor organizate în baze de date, în vederea rezolvării eficiente a problemelor informatice de gestiune
- *crearea tabelelor pentru un sistem de gestiune a unei bibliotecii, prin scrierea comenzilor DDL, respectând ordinea adecvată de definire a acestora* - *elaborarea unui set de comenzi SQL pentru analiza datelor unui sistem de rezervări hoteliere, prin generarea de rapoarte destinate managementului* - *crearea unei interogări avansate cu JOIN, GROUP BY, funcții agregate și condiții în HAVING, pentru a genera un raport personalizat pentru un lanț de restaurante prin identificarea sucursalelor cu vânzări peste medie* - *crearea unei baze de date care utilizează vederi, secvențe și indecși, pentru un sistem de rezervări* - *elaborarea unui set de comenzi SQL care gestionează tranzacții (BEGIN TRANSACTION, SAVEPOINT, COMMIT, ROLLBACK) și drepturi de acces pentru sistemul de procesare comenzi e-commerce, prin implementarea tranzacțiilor și tratare completă a erorilor* - *proiectarea unei aplicații Python care permite introducerea, modificarea și ștergerea datelor dintr-o bază de date (CRUD) pentru gestionarea inventarului unui magazin, implementând funcții separate create_produs(), read_produse(), update_pret() și delete_produs(), fiecare gestionând conexiunea, executând comanda SQL corespunzătoare (INSERT/SELECT/UPDATE/DELETE) și validând datele de intrare* - *elaborarea unui modul Python propriu care gestionează conexiunile și oferă funcții reutilizabile pentru execuția interogărilor pentru standardizarea accesului la baza de date în mai multe scripturi, creând clasa DatabaseManager cu metode connect(), execute_query(), execute_update() și close(), încapsulând logica de conexiune și tratare erori pentru reutilizare în diferite module* - *dezvoltarea unei aplicații complexe care folosește mai multe clase și metode (Connection, Cursor, commit, rollback) pentru operarea sigură a tranzacțiilor pentru un sistem bancar de transfer fonduri între conturi, implementând tranzacții atomice care debitează un cont și creditează altul, folosind commit() doar dacă ambele operații reușesc sau rollback() la eroare pentru menținerea consistenței datelor*
CS 6.5. Implementează aplicații care integrează în mod personalizat elemente specifice de limbaj de programare pentru prelucrarea datelor în învățarea automată, în vederea rezolvării eficiente a problemelor informatice
- *proiectarea unei aplicații Python complete care încarcă un fișier .csv, curăță datele, calculează statistici descriptive și afișează grafice multiple cu Matplotlib pentru analiza performanței sportive a elevilor dintr-un club sportiv, prin implementarea unui flux complet de activități* - *elaborarea unui mini-model de învățare automată pentru predicția mediei finale pe baza notelor la teste, utilizând clasele LinearRegression sau DecisionTreeClassifier din Scikit-learn și împărțirea datelor în seturi de antrenare și testare prin funcția train_test_split()* - *crearea unui mini-proiect de învățare automată care include preprocesarea datelor, documentând fiecare etapă, pentru un set de date privind prețurile locuințelor, având în vedere curățarea și completarea valorilor lipsă, codificarea variabilelor*

Clasa a XII-a
categorice, normalizarea valorilor numerice și antrenarea unui model simplu, cu documentarea fiecărei etape într-un notebook Python - *proiectarea unui pipeline complet de preprocesare a datelor folosind exclusiv NumPy, prin combinarea operațiilor de curățare, transformare și normalizare* - *modificarea unui cod, care folosește liste Python pentru a memora coordonatele a 1000 de puncte date și calculează distanțele euclidiene pentru toate perechile de puncte, pentru a îl optimiza cu operații vectorizate, utilizând biblioteca NumPy*

CONȚINUTURI ALE ÎNVĂȚĂRII

Clasa a XII-a

Domenii de conținut	Conținuturi
1. Organizarea conceptuală a datelor	1.1. Modelul conceptual relațional – entitate-relație - concepte de bază și caracteristici ale modelului conceptual entitate-relație: entitate (cu atribute opționale, obligatorii, volatile, respectiv cu identificatori unici, tipuri de identificatori unici – simplu/compus, natural/artificial), instanță, relație (opționalitate, cardinalitate, tipuri de relații din punctul de vedere al cardinalității, relații barate, relații ierarhice, relații recursive), supertip și subtip, transferabilitate, arc; - repere pentru identificarea entităților și reprezentarea convențională a diagramei entitate-relație (ERD), rezolvarea relațiilor M:M (mai mulți la mai mulți), reprezentarea istoricului datelor; - concepte de bază și caracteristici ale modelului fizic: tabelă (cu câmpuri, tipuri de date, cheie primară, chei externe, înregistrări, constrângeri, reguli de integritate - de entitate și referențială); - repere pentru transformarea modelului conceptual în model fizic (mapare). 1.2. Modele conceptuale de organizare a datelor folosite în învățarea automată - concepte de bază și caracteristici ale învățării automate: date, atribute, etichete, model, antrenare (învățare), testare, predicție; - caracteristici ale seturilor de date utilizate în învățarea automată: tipuri de date, date categorice, date complete/lipsă, dimensiune și calitate a datelor; - repere pentru transformarea datelor în formate utilizabile de către algoritmii de inteligență artificială (normalizare, scalare, augmentare, codificare a datelor, tratare a datelor lipsă, organizarea datelor în fișiere sau tabele pentru prelucrare automată).
2. Strategii de rezolvare a problemelor	2.1. Forme normale pentru proiectarea modelului conceptual al unei probleme de gestiune - caracteristici și avantaje ale primelor trei forme normale (FN1, FN2, FN3) și repere pentru utilizarea acestora în proiectarea modelului conceptual al unei probleme de gestiune. 2.2. Algoritmi utilizați în învățare automată - concepte de bază și caracteristici: învățare nesupervizată, clusterizare; - caracteristici ale algoritmului K-means, pentru învățarea nesupervizată; - concepte de bază și caracteristici: învățare supervizată, regresie, clasificare, arbore de decizie; - caracteristici ale algoritmului de regresie liniară, pentru învățare supervizată; - caracteristici ale algoritmului KNN (k Nearest Neighbours), pentru învățare supervizată; - caracteristici ale algoritmului de clasificare bazat pe arborele de decizie, pentru învățare supervizată; - caracteristici ale rețelelor neuronale, rețele perceptron, principiu de învățare prin rețeaua neuronală.
3. Memorarea datelor și organizarea codului în limbaj de programare	3.1. Comenzi SQL pentru gestionarea unei baze de date - caracteristici și sintaxă ale unor comenzi de bază pentru interogarea bazei de date (SQL): selecție, proiecție, ordonare, operatori, funcții scalare uzuale (de prelucrare a numerelor, a datelor calendaristice, a șirurilor de caractere, conversii), funcții agregate uzuale (pentru sumă, medie, numărare, minim, maxim, de tratare a valorii NULL), grupare, preluare a datelor din mai multe tabele (join), subinterogări; - caracteristici și sintaxă ale unor comenzi de bază pentru manipularea datelor (DML): inserare, modificare, ștergere; - caracteristici și sintaxă ale unor comenzi de bază pentru manipularea structurii bazei de date (DDL): creare a unei tabele, modificare a structurii unei tabele (adăugare a unui câmp, ștergere a unui câmp, modificare a unui câmp), ștergere a unei tabele, secvențe, indecși, vederi (view); - caracteristici și sintaxă ale unor comenzi de bază pentru gestionarea drepturilor de acces la baza de date (DCL): drepturi, acordare și retragere a unor drepturi; - caracteristici și sintaxă ale unor comenzi de bază pentru gestionarea tranzacțiilor din baza de date (TCL): inițiere, puncte de revenire, revenire, finalizare; - repere pentru executarea comenzilor;

Domenii de conținut	Conținuturi
	- repere pentru identificarea și utilizarea altor comenzi, adecvate pentru prelucrarea bazelor de date, în funcție de nevoile proprii. 3.2. Clase ale limbajului Python pentru interacțiunea cu bazele de date - caracteristici; - metode ale claselor principale (de exemplu, Connection, Cursor) pentru operații uzuale: conectare la baza de date, executare a unei comenzi, închidere a conexiunii cu baza de date; - repere pentru identificarea și utilizarea altor clase și metode adecvate pentru interacțiunea cu bazele de date, în funcție de nevoile proprii. 3.3. Biblioteci ale limbajului Python pentru învățare automată - caracteristici, clase și metode specifice din biblioteca Matplotlib pentru vizualizarea datelor: reprezentarea grafică a datelor prin puncte, grafic liniar, diagramă de dispersie, diagramă cu bare, histogramă, diagramă box-plot; - caracteristici, clase și metode specifice din biblioteca Pandas pentru curățarea, manipularea și analiza statistică a datelor: tabele (DataFrames), eliminarea valorilor extreme, obținere a valorilor pentru un atribut, conversii, statistici descriptive (medie, deviație standard, minim, maxim), calcule statistice de bază, determinarea corelațiilor între variabile, agregarea datelor pe categorii, frecvența valorilor unice; - caracteristici, clase și metode specifice din biblioteca Scikit-learn pentru rezolvarea problemelor de codificare și normalizare a datelor, clasificare, regresie și clusterizare: metode pentru antrenare, predicție și evaluare în rezolvarea problemelor de învățare automată; - caracteristici, clase și metode specifice din biblioteca NumPy pentru algebră liniară: operații cu liste de valori, operații cu matrici, rezolvarea sistemelor de ecuații; - repere pentru utilizarea algoritmilor prin metodele disponibile în biblioteci, pentru rezolvarea unor probleme de natură practică; - repere pentru identificarea și utilizarea altor clase și metode adecvate pentru învățare automată, în funcție de nevoile proprii.

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, cât și aplicativ, 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 calculatoarele 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 măsor? 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 clasei/școlii în care se desfășoară procesul didactic.

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

Activitățile de învățare alese trebuie să urmărească formarea competențelor din programă, precum și a unor competențe transversale, cum ar fi utilizarea tehnologiilor digitale pentru a aduce îmbunătățiri sau soluții noi pentru procese și produse, cu o abordare centrată pe factorul uman, prin implicarea individuală și colectivă în procesele de gândire critică și în utilizarea creativă și intenționată a tehnologiilor digitale, pentru a înțelege și a rezolva problemele conceptuale și situațiile problematice.
De asemenea, este necesar ca elevii să fie conștienți că trebuie să rămână informați cu privire la evoluțiile tehnologice digitale și la implicațiile lor în viața personală, profesională și în societale, să recunoască domeniile în care propria competență digitală trebuie îmbunătățită sau actualizată și să abordeze propriile nevoi în materie de competențe digitale în cadrul unui proces mai amplu de învățare pe tot parcursul vieții, de consolidare a capacităților și a autonomiei, oferind deschidere spre a îi sprijini și pe alții în dezvoltarea competențelor lor digitale.

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. 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.

Se recomandă îmbinarea metodelor didactice tradiționale de predare-învățare-evaluare (de exemplu, demonstrația, problematizarea, algoritmizarea, 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. Abordarea exploratorie, prin testare și corectare (trial and error) sprijină formarea gândirii critice și a capacității de autoînvățare.

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 algoritmii și corectează propriile erori. Se recomandă evaluări sumative/finale, după fiecare unitate de învățare.

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, rețea 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) studiați 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 metodele/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).

Pentru specializarea matematică-informatică, la clasele cu predarea disciplinei informatică în regim intensiv, **limbajul de programare de bază pentru implementare este Python**, secondat de limbajul C++, de exemplu, în una dintre variantele:
- în paralel (alocând pentru practică anumite ore, săptămânal, fiecărui limbaj, ca în cazul limbilor străine, sau exemplificând alternativ implementări în cele două limbaje), evidențiind pe parcurs asemănările și deosebirile dintre acestea;
- succesiv, realizându-se o recapitulare a unor concepte, în limbajul C++, conform programei școlare.
Predarea elementelor de limbaj specifice Python și C++ în paralel oferă o abordare solidă, pe baza unui plan didactic eficient, având în vedere consolidarea conceptelor comune și susținerea învățării prin comparație (observând asemănări de logică și structură, diferențe de sintaxă, o înțelegere mai profundă a conceptelor de bază, cum ar fi gestionarea automată sau manuală a memoriei), evidențierea complementarității conceptuale (Python oferă o sintaxă simplă, ușor de citit, care permite elevilor să se concentreze pe concepte fără să fie pus accent pe detalii tehnice, iar C++ facilitează înțelegerea profundă a mecanismelor din spatele limbajului — tipuri de date, gestionarea memoriei, pointeri, compilare), formarea unei gândiri flexibile în programare, evidențiindu-se faptul că limbajul este doar un instrument, iar gândirea logică este universală, facilitându-se astfel adaptabilitatea la orice alt limbaj viitor.

Predarea succesivă presupune ca elevii să utilizeze mai întâi limbajul Python, să stăpânească bazele, iar abia apoi să treacă la C++, pentru a aprofunda sau extinde conceptele deja învățate. Începând cu Python, elevii se concentrează pe logică, algoritmi și gândire procedurală, fără a pune accentul pe detalii tehnice, iar trecerea la C++ se face cu o bază solidă: ei știu deja să elaboreze algoritmi și să îi reprezinte în limbaj de programare, și învață doar cum se implementează aceștia în alt limbaj de programare, precum și aspecte noi, specifice acestui limbaj.
În exemplele de activități de învățare, limbajul de programare vizat este limbajul de bază studiat, Python, cu excepția cazului în care se precizează explicit limbajul C++.

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 schematică, 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.

Un exemplu, orientativ, de ordine de abordare a conținuturilor pentru clasa a XII-a, specializarea matematică-informatică, pentru clase cu predarea disciplinei *informatică* în regim intensiv, cu o abordare în paralel a celor două limbaje de programare

1. *Organizarea conceptuală a datelor. Model conceptual relațional – entitate-relație*
2. *Strategii de rezolvare a problemelor. Forme normale pentru proiectarea modelului conceptual al unei probleme de gestiune*
3. *Memorarea datelor și organizarea codului în limbaj de programare. Clase ale limbajului Python pentru interacțiunea cu bazele de date*
4. *Memorarea datelor și organizarea codului în limbaj de programare. Comenzi SQL pentru gestionarea unei bazei de date*
5. *Organizarea conceptuală a datelor. Modele conceptuale de organizare a datelor folosite în învățarea automată*
6. *Strategii de rezolvare a problemelor. Algoritmi utilizați în învățare automată*
7. *Memorarea datelor și organizarea codului în limbaj de programare. Biblioteci ale limbajului Python pentru învățare automată*

Se recomandă introducerea în tematica modelului entitate-relație prin exemple din viața reală (de exemplu, bibliotecă, magazin, școală) prin conectarea permanentă a conceptelor teoretice cu scenarii concrete, astfel încât elevii să înțeleagă conceptele specifice ca „entitate”, „relație”, „identificator unic”.
Procesul de învățare trebuie să fie centrat pe investigare și rezolvare de probleme, iar activitățile să favorizeze explorarea comparativă a soluțiilor. De exemplu, se poate analiza modul în care aceeași problemă de gestiune a datelor poate fi rezolvată atât prin comenzi SQL clasice, cât și prin instrumente moderne de analiză vizuală a datelor (de exemplu, Google Data Studio, Power BI).
Activitățile pot fi centrate pe proiecte de durată (individuale sau de grup) care implică proiectarea și gestionarea bazelor de date, dezvoltarea unor aplicații digitale, astfel încât elevii să poată experimenta întregul ciclu de dezvoltare al unei aplicații informatice – de la analiză și proiectare, până la testare și prezentarea rezultatelor.
Pentru clasa a XII-a, se recomandă ca elevii să valorifice competențele formate pe parcursul claselor anterioare în cadrul unor proiecte complexe, interdisciplinare, axate pe rezolvarea de probleme reale și pe construirea de produse informatice funcționale, integrând instrumente de lucru colaborativ online.
Se recomandă realizarea unui proiect integrator, care ar putea implica realizarea unei baze de date pentru un scenariu dat, în care să se integreze, pe parcurs, elemente din ce în ce mai complexe, pe baza studiului conținuturilor specifice, asigurând documentarea, identificarea necesarului afacerii, realizarea modelului conceptual (cu elemente cât mai variate), maparea acestuia, implementarea unor module ale acestuia, precum și testarea și realizarea documentației aferente. O astfel de integrare ar sublinia coerența practică a conținuturilor predate și ar antrena competențele transversale.
Se recomandă o abordare activă, practică și exploratorie, în care profesorul are rolul de mentor, iar elevul devine creator de soluții, astfel încât procesul de învățare să cultive autonomia, gândirea critică și etica în utilizarea tehnologiei. Profesorul ghidează procesul de învățare prin întrebări, studiu de caz și prin sprijin acordat în momentele-cheie, facilitând trecerea de la învățarea asistată la învățarea autonomă și reflexivă.
Se recomandă integrarea noțiunilor introductive de inteligență artificială în proiecte interdisciplinare. Elevii pot explora modul în care o bază de date alimentată cu exemple reale poate susține un model de învățare automată, pot reflecta critic asupra modului în care AI influențează deciziile, discutând despre biasul datelor, confidențialitate și responsabilitate în contextul discuțiilor despre algoritmii utilizați în învățarea automată.
Se recomandă ca, la finalul clasei a XII-a, profesorii să discute și despre perspectivele legate de competențele specifice programării, încurajând elevii să utilizeze competențele dobândite în cadrul unor proiecte de portofoliu, pentru admitere la facultate sau angajare. De exemplu, produsul software realizat la clasă poate fi perfecționat și prezentat la un concurs sau interviu. Acest tip de conexiune cu lumea reală poate crește motivația elevilor și evidenția eficiența curriculară, prin dimensiunea formării profesionale.

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

Pentru C++:

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

Pentru Python și C++:

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

Instrumente online pentru implementarea soluțiilor

- Online GDB (https://www.onlinegdb.com);
- Programiz (https://www.programiz.com/);
- w3schools (https://www.w3schools.com/);
- Repl.it (https://replit.com/) - inclusiv pentru activități de programare colaborativă;
- Jupyter Notebooks (https://jupyter.org/, https://colab.google/) - inclusiv pentru activități de programare colaborativă;
- Raptor (https://raptor.martincarlisle.com/) - pentru reprezentare sub formă de schemă logică;
- 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 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, inteligență artificială și proiecte interactive;
- Wolfram ALPHA (https://www.wolframalpha.com/) platformă de inteligență computațională care facilitează învățarea diverselor discipline;
- UiPath Foundation (https://aigeneration.uipathfoundation.com/ro/), platformă ce conține planuri de lecție, prezentări, fișe, teste privind introducerea în inteligența artificială;
- pyML.ro (https://www.pyml.ro/), platformă cu lecții pentru introducere în învățare automată.

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.;
- Brainbashers (https://www.brainbashers.com/queens.asp) - 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;
- CS Academy (https://csacademy.com/app/graph_editor/) - 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, atribute, 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/) pentru a antrena modele simple (clasificare de imagini, recunoaștere de sunete), conectând activitățile la domenii diverse (biologie, arte, științe sociale);
- 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,

Nume și prenume	Grad didactic/Titlu științific	Instituție de apartenență, localitate, județ
		Facultatea de Automatică și Calculatoare
NIȚU Alina	profesor, gradul didactic I	Liceul Teoretic „Ovidius”, Constanța, județul Constanța
NURLA Aidan	profesor, gradul didactic I	Colegiul Naţional Militar „Alexandru Ioan Cuza”, Constanţa, județul Constanța
OANCEA Romana	conferențiar universitar, doctor	Academia Forţelor 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 Naţional Militar „Dimitrie Cantemir”, Breaza, județul Prahova
TEGLAȘ Maria	profesor, gradul didactic II	Colegiul Naţional 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