PLANIFICARE CALENDARISTICĂ ANUALĂ

Model orientativ pentru manual • An școlar 2026–2027

Disciplina: Informatică • Clasa a IX-a • Curriculum de specialitate (CS), matematică-informatică • 2 ore/săptămână (teorie și aplicații practice integrate)

Model general • 34 de săptămâni de activități • 68 ore

Repartizarea orientativă a orelor

Modul	Capitol / activitate	Ore disciplină	Ore R incluse	Alte activități
I	Gândirea computațională. Strategii de rezolvare a problemelor	12	3	—
II	Prelucrări numerice. Algoritmi elementari. Baze de numerație	16	4	—
III	Subprograme. Introducere în programarea orientată pe obiecte	12	3	—
IV	Modalități de comunicare cu programul. Fișiere text. Interfețe grafice	14	4	—
V	Modelul conceptual liniar. Clasa list. Metode de sortare. Eficiența algoritmilor	14	3	—
TOTAL	5 capitole	68	17	—

Din cele 68 de ore de informatică, 51 de ore sunt destinate predării, aplicării și evaluării, iar 17 ore sunt rezervă curriculară R (25%), pentru consolidare, aprofundare, diferențiere și activități remediale. În săptămânile marcate R: 1 oră pentru conținuturi și aplicații curente și 1 oră R; în celelalte săptămâni: 2 ore pentru predare, aplicații și evaluare.

Intervalele pentru „Săptămâna verde”, „Școala altfel” și vacanța mobilă se stabilesc conform calendarului aplicabil fiecărei unități de învățământ. Repartizarea pe săptămâni este orientativă și se adaptează orarului.

Planificarea materiei pe săptămâni

Modul / perioadă	Unitatea de învățare	Competențe specifice	Conținuturi / activități	Nr. ore	Săptă mâna	Observații
M I	Gândirea computațională. Strategii de rezolvare a problemelor	CS 1.3, 2.3, 3.3	**I.1. Gândirea computațională** • Definiția gândirii computaționale; avantajul aplicării în multiple domenii (educație, cercetare, medicină, economie, industrie) • Cele patru caracteristici: descompunerea, recunoașterea tiparelor, abstractizarea, algoritmizarea • Studiu de caz: semafoare inteligente (mapare pe cele 4 caracteristici) • Activități: analiză de situații cotidiene; proiect de echipă (cele 4 componente pe o situație din viața reală); portofoliu digital	2	S1	2 ore (predare/aplicații/evaluare)
M I	Gândirea computațională. Strategii de rezolvare a problemelor	CS 1.3, 2.3, 4.3	**I.2. Principalele moduri de reprezentare a algoritmilor** • Noțiunea de algoritm; proprietăți: generalitate, corectitudine, finitudine, claritate, eficiență • Exemplu cotidian: împrumutul unei cărți de la bibliotecă (pași ordonați; observarea rolului fiecărui pas) • Blocurile unei scheme logice • Elementele principale ale limbajului pseudocod • Clasificarea limbajelor de programare: nivel scăzut / înalt; compilate / interpretate; • Tabel comparativ (schema logică / pseudocod / limbaj interpretat / compilat) și concluzii • Activități: reprezentări ale unor algoritmi folosind schema logică și limbajul pseudocod; portofoliu digital	2	S2	2 ore (1 oră R)
M I	Gândirea computațională. Strategii de rezolvare a problemelor	CS 2.3, 3.3, 4.3	**I.3. Scheme logice și limbajul pseudocod** • Rolul schemelor logice și al pseudocodului în înțelegerea, analiza și transpunerea algoritmilor • Aplicații practice • Activități: scriere în paralel pseudocod + schemă liniară;	2	S3	2 ore (predare/aplicații/evaluare)
M I	Gândirea computațională. Strategii de rezolvare a problemelor	CS 3.3, 4.3, 5.3	**I.3. Scheme logice și limbajul pseudocod — aplicații (continuare)** • Aplicații practice • Transpuneri schemă logică ↔ pseudocod; • Portofoliu digital • Activități R: consolidare	2	S4	2 ore (1 oră R)
M I	Gândirea computațională. Strategii de rezolvare a problemelor	CS 3.3, 5.3	**I.4. Testarea algoritmilor. Urmărirea evoluției valorilor variabilelor** • De ce testăm: corectitudine, completitudine, robustețe; acoperirea cerințelor • Criterii de testare: cazuri generale; cazuri limită; cazuri speciale; valori invalide/exceptate; acoperirea tuturor ramurilor decizionale • Testarea structurilor repetitive: zero executări / o executare / N executări • Urmărirea evoluției variabilelor (tabele de evoluție a valorilor variabilelor) • Activități: proiectarea unui set de teste pentru un algoritm dat și depanarea sa	2	S5	2 ore (predare/aplicații/evaluare)
M I	Gândirea computațională. Strategii de rezolvare a problemelor	CS 2.3, 3.3, 5.3	**I.5. Principalele etape ale elaborării unui program; recapitulare și evaluare**	2	S6	2 ore (1 oră R)

			• Etape: analiza problemei; proiectarea soluției; implementarea; testarea; depanarea • Descrierea etapelor elaborării unui program • Evaluare formativă/sumativă pe competențele CS 1.3–5.3; feedback • Activități R: probleme propuse / fișă de evaluare capitolul I; remediere			
M II	Prelucrări numerice. Algoritmi elementari. Baze de numerație	CS 1.2, 2.2, 3.2	**II.1. Accesarea cifrelor unui număr** • Tipuri de date și variabile în Python; operatori aritmetici • Parcurgerea cifrelor unui număr natural •Construirea unor rezultate pe baza cifrelor • Activități: aplicații	2	S7	2 ore (predare/aplicații/evaluare)
M II	Prelucrări numerice. Algoritmi elementari. Baze de numerație	CS 2.2, 3.2	**II.2. Algoritmi de construire a numerelor prin adăugarea cifrelor la dreapta** • Adăugarea unei cifre; adăugarea succesivă a mai multor cifre • Adăugare condiționată: if; if…else; if…elif…else; match (alegerea cifrei c) • Operatori relaționali • Activități: aplicații; oră R pentru consolidare	2	S8	2 ore (1 oră R)
M II	Prelucrări numerice. Algoritmi elementari. Baze de numerație	CS 2.2, 3.2	**II.3. Algoritmi de construire a numerelor prin adăugarea cifrelor la stânga** • Ideea păstrării ordinii cifrelor la inserarea în stânga; puteri ale lui 10 • Operatori logici • Structuri repetitive (while) pentru prelucrarea cifrelor; • Aplicații: sumă a cifrelor; variante de construire a unui număr nou • Activități: exerciții cu operatori logici în condiții compuse	2	S9	2 ore (predare/aplicații/evaluare)
M II	Prelucrări numerice. Algoritmi elementari. Baze de numerație	CS 3.2, 4.2	**II.4. Divizorii unui număr natural** • Noțiunea de divizor; numărul și suma divizorilor • Instrucțiunea for și funcția range (start, stop, pas) — reguli și greșeli frecvente • Testarea primalității; aplicații pe divizori • Activități R: probleme de numărare/calculul sumei divizorilor; optimizări elementare de interval	2	S10	2 ore (1 oră R)
M II	Prelucrări numerice. Algoritmi elementari. Baze de numerație	CS 3.2, 4.2	**II.5. Prelucrarea selectivă a cifrelor unui număr** • Selecția cifrelor după proprietăți (pare/impare, aparținând unui interval etc.) • Instrucțiunile break și continue • Construirea unui rezultat din cifrele selectate; aplicații cu șiruri de valori • Activități: probleme de filtrare; oprirea anticipată a unei bucle	2	S11	2 ore (predare/aplicații/evaluare)
M II	Prelucrări numerice. Algoritmi elementari. Baze de numerație	CS 3.2, 6.2	**II.6. Algoritmi elementari pentru divizori și multipli comuni** • Descompunerea în factori primi • CMMDC • algoritmi elementari pentru determinarea celui mai mare divizor comun (algoritmul lui Euclid cu scăderi repetate, respectiv cu împărțiri repetate) • CMMMC; relația cu CMMDC; multiplii unui număr • Activități R: aplicații combinate (factori primi + CMMDC/CMMMC)	2	S12	2 ore (1 oră R)
M II	Prelucrări numerice. Algoritmi elementari. Baze de numerație	CS 2.2, 3.2	**II.7. Baze de numerație. Conversii între sisteme de numerație** • Sisteme de numerație • Conversie din baza 10 în baza 2 • Conversie dintr-o bază b în baza 10 • Activități: conversii ghidate; probleme de validare a scrierii într-o bază	2	S13	2 ore (predare/aplicații/evaluare)

M II	Prelucrări numerice. Algoritmi elementari. Baze de numerație	CS 3.2, 4.2, 6.2	**Probleme propuse și fișă de evaluare — Capitolul II** • Aplicații integrate pe prelucrări numerice (cifre, divizori, baze) • Fișă de evaluare de capitol; barem / feedback • Activități R: remediere pe itemii cu erori frecvente; aprofundare selectivă • Evaluare formativă/sumativă	2	S14	2 ore (1 oră R)
M III	Subprograme. Introducere în programarea orientată pe obiecte	CS 1.5, 2.5, 3.3, 3.5, 6.3	**III.1. Proiectarea modulară a algoritmilor. Funcții fără parametri** • De ce modularizăm: reutilizare, depanare, lucru în echipă • Definirea și apelul unei funcții fără parametri; instrucțiunea return • Diferența dintre funcții care afișează și funcții care returnează o valoare • Activități: funcții simple (par/impar, oglindit, nr. cifre, sumă divizori)	2	S15	2 ore (predare/aplicații/evaluare)
M III	Subprograme. Introducere în programarea orientată pe obiecte	CS 1.5, 2.5, 3.5	**III.2. Funcții cu parametri (partea I)** • Antetul funcției; parametri formali vs. parametri efectivi • Apel pozițional și apel prin nume; valori returnate • Aplicații • Activități R: scriere și testare de funcții	2	S16	2 ore (1 oră R)
M III	Subprograme. Introducere în programarea orientată pe obiecte	CS 2.5, 4.5, 6.5	**III.2. Variabile locale și globale; mecanismul apelului (partea II)** • Vizibilitatea variabilelor: locale vs. globale • Parametri cu valori implicite • Mecanismul de executare al unei funcții (stivă de apeluri — la nivel introductiv) • Activități: analiza secvențelor; corectarea erorilor de vizibilitate	2	S17	2 ore (predare/aplicații/evaluare)
M III	Subprograme. Introducere în programarea orientată pe obiecte	CS 3.5, 4.5	**III.3. Module. Funcții predefinite** • import al unui modul; import selectiv; apel modul.funcție • Module și funcții predefinite utile (math, fractions) • caracteristici și repere de utilizare a unor subprograme predefinite în Python pentru operații matematice uzuale (de exemplu radical, valoare absolută, parte întreagă, rotunjire) și conversii • Funcții care returnează mai multe valori • Activități R: mini-aplicații cu import; documentare minimală a unei funcții	2	S18	2 ore (1 oră R)
M III	Subprograme. Introducere în programarea orientată pe obiecte	CS 1.5, 2.5, 4.5	**III.4. Introducere în programarea orientată pe obiecte** • Clasă, obiect, instanță; atribute și metode (exemple: Elev, int, Fraction) • Crearea și utilizarea obiectelor; metode tipice • Biblioteci / tipuri predefinite ca exemple de clase • Activități: explorare Fraction / float; exerciții de identificare clasă vs. obiect	2	S19	2 ore (predare/aplicații/evaluare)
M III	Subprograme. Introducere în programarea orientată pe obiecte	CS 3.5, 5.5, 6.3, 6.5	**Probleme propuse și fișă de evaluare — Capitolul III** • Aplicații modulare (funcții + module + elemente OOP introductive) • Evaluare formativă/sumativă • Activități R: proiect scurt modular; remediere	2	S20	2 ore (1 oră R)
M IV	Modalități de comunicare cu programul. Fișiere text. Interfețe grafice	CS 1.4, 2.4, 3.4	**IV.1. Gestionarea comunicării cu utilizatorul. Principii de lucru cu fișiere text** • Comunicare prin consolă vs. fișiere vs. interfață • Utilitatea fișierelor text; principii de bază; clasa TextIOWrapper • Deschidere/închidere; moduri de acces; • Citire/scriere elementară	2	S21	2 ore (1 oră R)

			• Tratarea excepțiilor • Activități R: exerciții open/read/write			
M IV	Modalități de comunicare cu programul. Fișiere text. Interfețe grafice	CS 3.4, 4.4, 6.4	**IV.2. Aplicații practice cu fișiere text** • Programe complete: citire din fișier, prelucrare, scriere rezultate • Exemplu tip Fibonacci scris în fișier; variabile de stare • Tratarea erorilor la deschidere / format greșit • Activități: aplicații	2	S22	2 ore (predare/aplicații/evaluare)
M IV	Modalități de comunicare cu programul. Fișiere text. Interfețe grafice	CS 3.4, 5.4, 6.4	**IV.2. Probleme cu fișiere text (continuare / aprofundare)** • Prelucrări numerice din fișier; validare; fișiere de intrare/ieșire • Probleme tip: medii elevi, încasări pe zile, curs valutar • Activități R: depanare pe fișiere reale de test; diferențiere • Evaluare formativă/sumativă	2	S23	2 ore (1 oră R)
M IV	Modalități de comunicare cu programul. Fișiere text. Interfețe grafice	CS 1.4, 3.4, 6.4	**IV.3. Realizarea interfețelor grafice utilizând biblioteca Tkinter** • Import tkinter; fereastra principală (Tk): title, geometry, mainloop • Widget-uri: Label, Button; metoda pack; evenimente (command) • Afișarea mesajelor: messagebox • Activități: aplicații practice	2	S24	2 ore (predare/aplicații/evaluare)
M IV	Modalități de comunicare cu programul. Fișiere text. Interfețe grafice	CS 3.4, 4.4, 6.4	**IV.4. Componente pentru introducerea datelor (Tkinter)** • Entry / Text: preluarea datelor (get), conversii, afișare rezultate • Validare și mesaje către utilizator • Activități R: mini-formular	2	S25	2 ore (1 oră R)
M IV	Modalități de comunicare cu programul. Fișiere text. Interfețe grafice	CS 3.4, 4.4, 6.4	**IV.5. Manageri de geometrie. Canvas** • Frame; pack / grid / place • Canvas: desenarea elementelor grafice simple • Activități: aplicații practice	2	S26	2 ore (predare/aplicații/evaluare)
M IV	Modalități de comunicare cu programul. Fișiere text. Interfețe grafice	CS 3.4, 5.4, 6.4	**Probleme propuse și fișă de evaluare — Capitolul IV** • Miniaplicații integrate (fișiere ± Tkinter) • Evaluare formativă/sumativă • Activități R: proiect	2	S27	2 ore (1 oră R)
M V	Modelul conceptual liniar. Clasa list. Metode de sortare. Eficiența algoritmilor	CS 1.1, 2.1, 3.1, 5.1	**V.1. Modelul conceptual liniar: listă, stivă, coadă și listă de frecvențe** • Definiția și caracteristicile unei liste; ordine, lungime, acces • Acces direct vs. acces secvențial; parcurgere • Cazuri particulare: stivă (LIFO) și coadă (FIFO) • Lista de frecvențe — utilitate pe domenii restrânse de valori • repere pentru parcurgerea elementelor și pentru aplicarea algoritmilor de bază pentru prelucrarea datelor organizate liniar, cu sau fără memorare • Activități: aplicații	2	S28	2 ore (predare/aplicații/evaluare)
M V	Modelul conceptual liniar. Clasa list. Metode de sortare. Eficiența algoritmilor	CS 1.4, 2.4, 3.4	**V.2. Clasa list. Accesarea elementelor. Operatori specifici** • Caracteristicile tipului list în Python; creare prin constante • Indexare; slice-uri; operatori (apartenență, concatenare, multiplicare, relaționali) • Greșeli frecvente la indexare / slice-uri • Activități R: aplicații	2	S29	2 ore (1 oră R)

M V	Modelul conceptual liniar. Clasa list. Metode de sortare. Eficiența algoritmilor	CS 3.1, 3.5, 6.1	**V.3. Crearea unei liste prin citire. Funcții predefinite pentru liste** • Citirea a n elemente într-o listă; parcurgeri • Funcții predefinite: len, min, max, sum, sorted — comportament și utilizare • Exemplu: maxim și medie pe o listă • Activități: aplicații; greșeli tipice (sorted vs. sort)	2	S30	2 ore (predare/aplicații/evaluare)
M V	Modelul conceptual liniar. Clasa list. Metode de sortare. Eficiența algoritmilor	CS 3.2, 3.4, 6.4	**V.4–V.5. Metode ale clasei list; generare sistematică** • Adăugare / inserare / ștergere / căutare: append, insert, remove, pop, index, count, copy, etc • Generarea sistematică a elementelor • Activități R: construirea listelor pas cu pas; căutări și actualizări	2	S31	2 ore (1 oră R)
M V	Modelul conceptual liniar. Clasa list. Metode de sortare. Eficiența algoritmilor	CS 3.2, 4.2, 5.2, 6.2	**V.6. Metode de sortare a elementelor unei liste** • sort / sorted — diferențe (in-place vs. listă nouă) • Sortarea prin selecție; sortarea prin metoda bulelor (Bubble Sort) • Activități: aplicații practice	2	S32	2 ore (predare/aplicații/evaluare)
M V	Modelul conceptual liniar. Clasa list. Metode de sortare. Eficiența algoritmilor	CS 3.1, 5.2, 6.1	**V.7. Liste de frecvențe. Sortarea cu listă de frecvențe** • Construirea unei liste de frecvențe pe un domeniu restrâns • Sortare / reordonare pe baza frecvențelor • Activități R: probleme de numărare și afișare după frecvență; consolidare	2	S33	2 ore (1 oră R)
M V	Modelul conceptual liniar. Clasa list. Metode de sortare. Eficiența algoritmilor	CS 4.1, 5.2, 5.3, 6.1	**V.8. Eficiența algoritmilor** • Ideea de ordine de complexitate • eficiența algoritmilor: caracteristici ale eficienței din punctul de vedere al spațiului de memorie și din punctul de vedere al timpului de executare, criterii pentru determinarea ordinului de complexitate a unui algoritm polinomial (notația O) • Compararea unor soluții pe același tip de problemă • Probleme propuse / fișă de evaluare — capitolul V și sinteză anuală • Evaluare finală și feedback	2	S34	2 ore (predare/aplicații/evaluare)

R = activitate din rezerva curriculară (consolidare, aprofundare, recuperarea lacunelor, investigație, proiect). Orele de teorie și aplicațiile practice se integrează în cele 2 ore săptămânale. Numerotarea S1–S34 este orientativă.