Introducere în programare

Project number : 2023-2-IT03-KA220-YOU-000179130

Co-funded by
the European Union

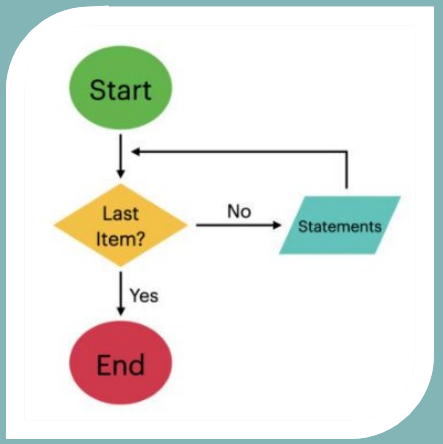
Concepte esențiale

1. Modularitate și Reutilizabilitate
2. Lizibilitate și Sustenabilitate
3. Abstractizare
4. Prevenirea Erorilor
5. Eficiență
6. Adaptabilitate
7. Învățare și Rezolvare a Problemelor

Controlul fluxului programului

„if-else”: permite programului să ia decizii.
„if” – dacă o condiție este adevărată → execută o acțiune;
„else” → execută o acțiune diferită.

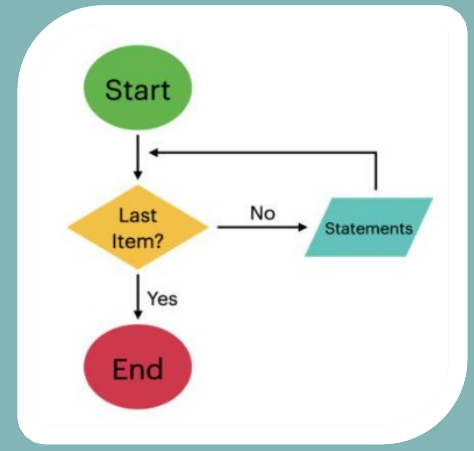
Bucle (Loops)
Permit repetarea unui set de instrucțiuni de un anumit număr de ori sau până când este îndeplinită o anumită condiție.



DEAF YOUNG CODE

Controlul fluxului programului

For Loop (Bucla for): Folosită pentru a repeta instrucțiuni de un număr specific de ori, gestionând adesea o variabilă contor.
While Loop (Bucla while): Repetă un bloc de cod atât timp cât o anumită condiție rămâne adevărată.
Do-While Loop (Bucla do-while): Similară cu bucla while, dar garantează că blocul de cod este executat cel puțin o dată, verificând condiția doar la final.



Video

Algoritmi – Pași clari către obiectiv

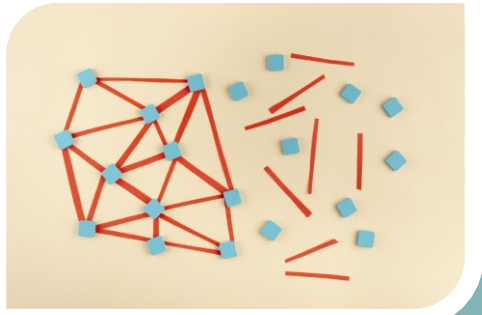
DEAF YOUNG CODE

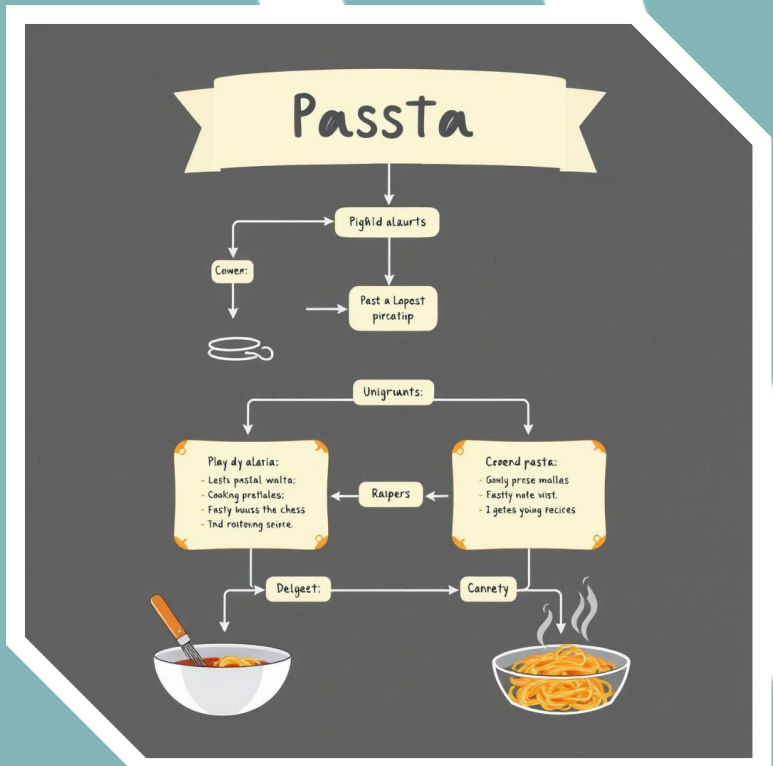
Ce este un algoritm?

O secvență de pași bine definiți, concepuți pentru a rezolva o problemă specifică.

Este ca un set de instrucțiuni pas cu pas.

Este ca o rețetă, dar pentru calculatoare!





Activitate

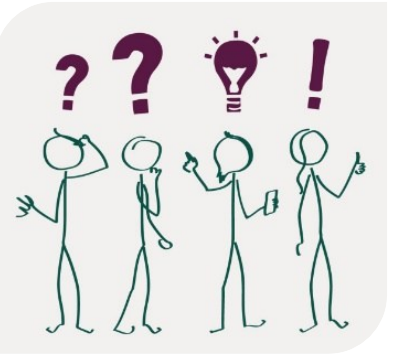
Algoritmi

DEAF YOUNG CODE

Exercițiu: Crearea de pseudocod sau diagrame de flux

Activitate

FlowChart

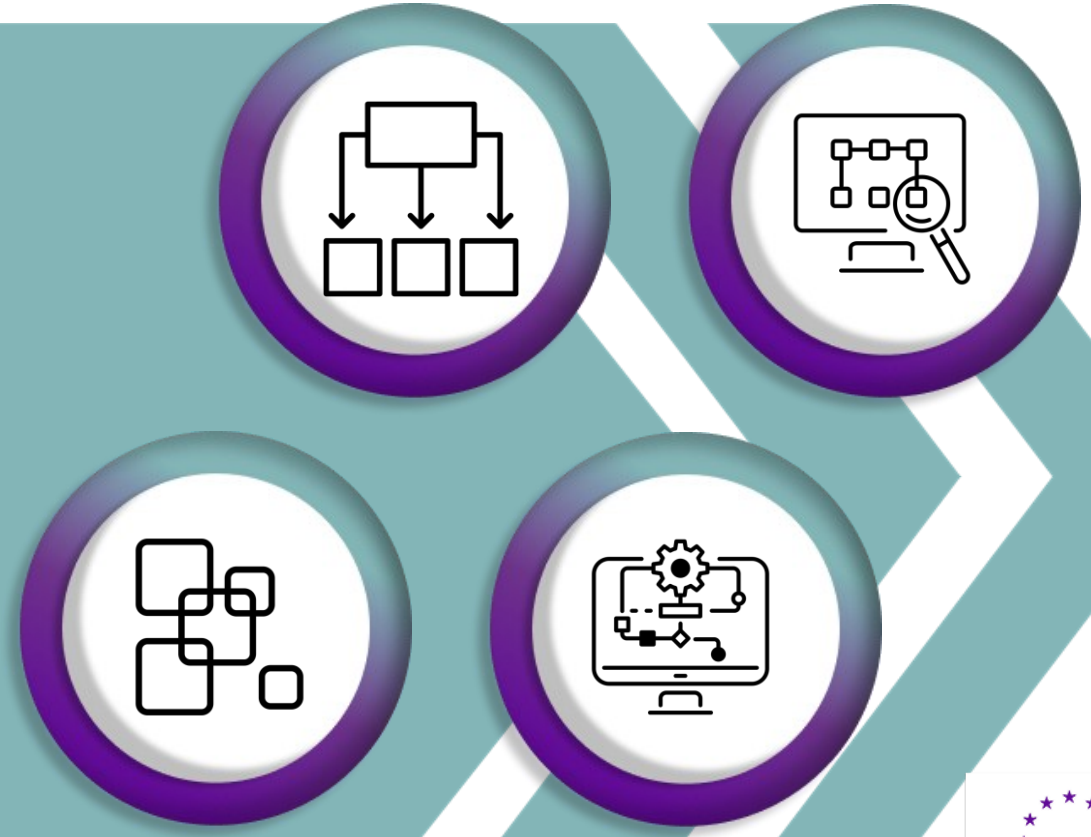


Exemple din viața reală

1. Prepararea cafelei
2. A bea apă de la robinet
3. A te îmbrăca înainte pentru a ieși afară
4. A lua autobuzul
5. Spălatul dinților cu pastă de dinți
6. Spălatul vaselor cu detergent de vase
7. Încărcarea telefonului mobil

Algoritmi – pas cu pas

1. Rezolvarea problemelor
2. Eficiență
3. Automatizare
4. Fundamente



Co-funded by the European Union

Algoritmi – pas cu pas

DEAF YOUNG CODE

Rezolvarea problemelor

Împarte problemele complexe în părți mai mici și mai ușor de gestionat.

Eficiență

Un algoritm bun rezolvă o problemă rapid, folosind cât mai puține resurse.

Automatizarea

Permite computerelor să execute sarcini automat, fără intervenție umană.

Fundamente

Înțelegerea algoritmilor formează baza pentru scrierea unui cod eficient și corect.

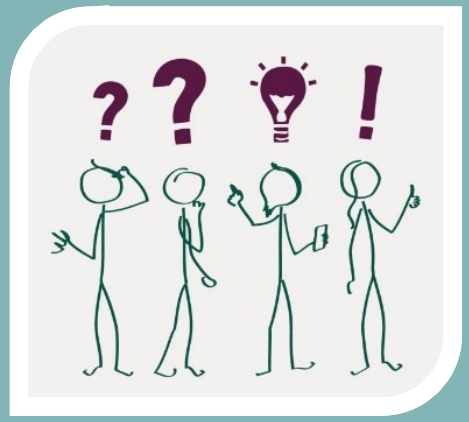
Activitate

Algoritmi

DEAF YOUNG CODE

Exemple din viața reală

Motoarele de căutare, navigația GPS și sistemele de recomandare se bazează pe algoritmi.



Activitate

Primul tău program

DEAF YOUNG CODE

Alegerea mediului și a IDE-ului tău

Python IDLE: Un mediu bazat pe text pentru scrierea efectivă a codului Python.

Structura de bază a unui program (Basic Program Structure)

Instrucțiunea Print (Print Statement): Afișează text pe ecran (de exemplu: `print("Hello, World!")`).

Logică de bază (Basic Logic): Efectuarea de calcule simple sau comparații.

Activitate

Primul tău program

Print (Hello Bucuresti)
1+1
1+3*4
Print(Your name)

Primul tău program

Alegerea mediului (Choosing Your Environment)
Scratch: Blocuri de tip „drag-and-drop” care fac programarea distractivă și vizuală. Ideal pentru începători!
Code.org: Tutoriale interactive care te ghidează pas cu pas, învățând concepte fundamentale.
Python IDLE: Un mediu bazat pe text pentru scrierea efectivă a codului Python.

Structura de bază a unui program (Basic Program Structure)
Instrucțiunea Print (Print Statement): Afișează text pe ecran (de exemplu: print("Hello, World!")).
Input: Permite programului să primească informații de la utilizator (de exemplu: name = input("Cum te numești? ")).
Logică de bază (Basic Logic): Efectuarea de calcule simple sau comparații.

Activitate

Video

Co-funded by the European Union

Depanarea (Debugging) – Găsirea și corectarea erorilor

DEAF YOUNG CODE

Depanarea (Debugging) este procesul de găsire și rezolvare a erorilor (bug-uri) din codul tău.

Este o abilitate esențială pentru orice programator.

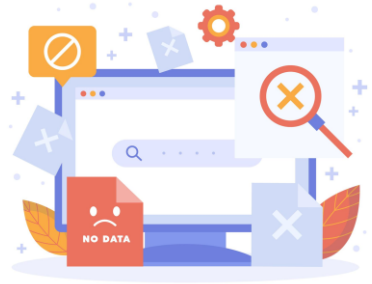
.

Surse comune de erori (Common Sources of Errors)

Erori de sintaxă (Syntax Errors)
Greșeli în regulile limbajului de programare (de exemplu: greșeli de scriere, lipsa punctuației).

Erori de rulare (Runtime Errors)
Erori care apar în timpul execuției programului (de exemplu: împărțire la zero, accesarea unui fișier inexistent).

Erori de logică (Logic Errors)
Programul rulează, dar nu produce rezultatele așteptate (de exemplu: calcule incorecte, condiții greșite).



Depanarea (Debugging) – Găsirea și corectarea erorilor

DEAF YOUNG CODE

Sfaturi pentru depanare

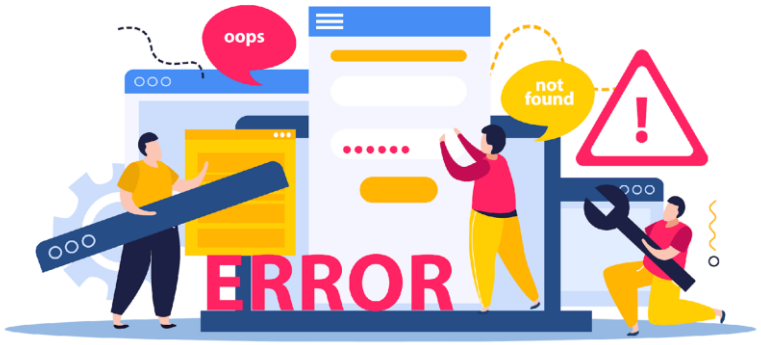
Citește mesajele de eroare:
Acordă atenție mesajului de eroare. De multe ori oferă indicii despre problemă.

Folosește un debugger:
Debugger-ele sunt instrumente care îți permit să parcurgi codul linie cu linie și să verifici valorile variabilelor.

Instrucțiuni print (print statements):
Adaugă instrucțiuni `print` pentru a afișa valorile variabilelor în diferite puncte din cod.

Testează-ți codul:
Testează programul cu diferite intrări pentru a descoperi cazuri limită care pot cauza erori.

Debugging cu rața de cauciuc (Rubber Duck Debugging):
Explică-ți codul linie cu linie unei rațe de cauciuc (sau oricărui obiect neînsuflețit). Acest lucru te poate ajuta să identifici erori de logică.

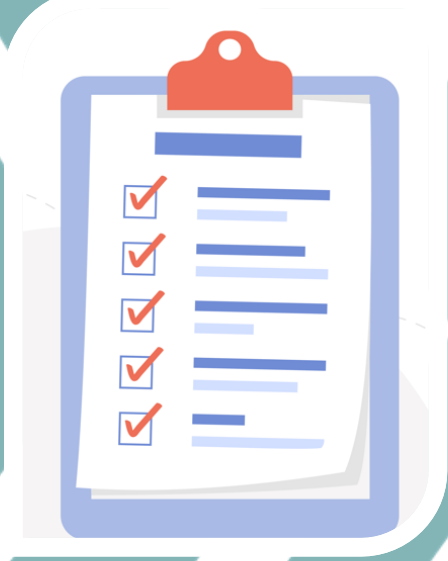


Recapitularea modulului

DEAF YOUNG CODE

Concepte cheie revizuite :

- Definiția codării și programării
- Înțelegerea variabilelor și a tipurilor de date
- Utilizarea operatorilor și a expresiilor
- Controlul fluxului programului cu instrucțiuni if și bucle
- Importanța algoritmilor în rezolvarea problemelor
- Tehnici de bază pentru depanare (debugging)

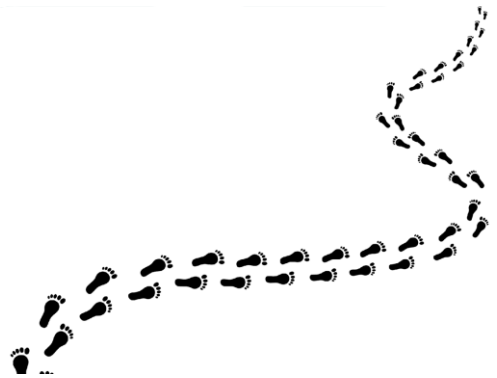


Pașii următori

DEAF YOUNG CODE

Privind înainte (Looking Ahead)

- Explorarea unor structuri de date mai avansate (liste, dicționare etc.)
- Învățarea modului de a crea funcții și module
- Construirea unor programe și proiecte mai complexe
- Aprofundarea unor limbaje de programare specifice



Activitate

Întrebări

Întrebarea 1

Ce este un program?

A) Un tip de hardware folosit în calculatoare.
B) Un joc jucat pe un computer.
C) Un set de instrucțiuni pe care un computer le urmează pentru a îndeplini o sarcină.
D) Un dispozitiv care stochează date.

Întrebarea 2

Care dintre următoarele este un exemplu de variabilă în programare?

A) O comandă pentru afișarea textului pe ecran.
B) O locație de stocare care reține date, cum ar fi un număr sau un text.
C) Un mesaj care explică ce face codul.
D) Un tip de limbaj de programare.

Întrebarea 3

Ce face o buclă (loop) într-un program?

A) Verifică dacă o condiție este adevărată sau falsă.
B) Repetă un set de instrucțiuni de mai multe ori.
C) Stochează date pentru utilizare ulterioară.
D) Trimite date către un alt computer.

Întrebarea 4

De ce este important debugging-ul în programare?

A) Pentru a adăuga funcționalități noi programului.
B) Pentru a identifica și corecta erorile (bug-urile) din cod.
C) Pentru a face programul să ruleze mai rapid.
D) Pentru a traduce codul în alte limbaje.

Mulțumim pentru atenție!

DEAF YOUNG CODE

Stai aproape pentru următoarea prezentare!

Project number : 2023-2-IT03-KA220-YOU-000179130

Co-funded by the European Union

LICEUL TEHNOLOGIC SPECIAL „VASILE PAVELCU“ - IAȘI