PROIECT DE LECȚIE

Componenta introductivă
Profesor: Ungurean Cătălina- Iuliana
Data: 25.05.2025
Clasa: a XI-a
Profil/Specializare: : Matematică-informatică intensiv informatică
Disciplina: Informatică

Unitatea de învățare: Grafuri neorientate și grafuri orientate
Lecția: Parcurgerea grafurilor în lățime și în adâncime
Tipul lecției: Transmitere și asimilare de noi cunoștințe

Competențe generale/ Obiective cadru: Identificarea datelor care intervin într-o problemă și aplicarea algoritmilor fundamentali de prelucrare a acestora

Competențe specifice/ Obiective de referință:

- Transpunerea unei probleme din limbaj natural în limbaj de grafuri, folosind corect terminologia specifică
- Analizarea unei probleme în scopul identificării datelor necesare și alegerea modalităților adecvate de structurare a datelor care intervin într-o problemă
- Descrierea algoritmilor fundamentali de prelucrare a grafurilor și implementarea acestora într-un limbaj de programare
- Analizarea în mod comparativ a avantajelor utilizării diferitelor metode de structurare a datelor necesare pentru rezolvarea unei probleme
- Aplicarea în mod creativ a algoritmilor fundamentali în rezolvarea unor probleme concrete

Obiective:
La sfârșitul lecției, elevul va fi capabil să:

- **O1:** Explice conceptele de parcurgere în lățime (BFS) și în adâncime (DFS) în cadrul grafurilor.
- **O2:** Aplice algoritmii BFS și DFS pe grafuri neorientate/orientate.
- **O3:** Implementeze în C++ algoritmii BFS și DFS pentru un graf dat.

Strategia didactică:

- **Metode:** conversația, explicația, demonstrația, exercițiul, învățarea prin descoperire
- **Mijloace de învățământ:** videoproiector, tablă, markere, laptopuri, editor de cod (ex. Code::Blocks), fișe de lucru, temă pentru acasă
- **Forme de organizare a activității:** frontală, pe grupe, individuală (la calculator)

Evaluare:

- Verificarea orală a răspunsurilor

- Evaluare practică: rularea corectă a algoritmilor implementați în C++
- Fișă de lucru cu itemi de completare și aplicare
- Temă pentru acasă si verificarea orală a acesteia

Bibliografie:

- Manual de informatică – clasa a XI-a
- Documentația oficială C++

Desfășurarea lecției/ activitații didactice

Evenimentele lecției	**Activitățile din lecție**	**Strategii didactice**
Captarea atenției	Verificarea prezenței. **Provocarea orei**: „V-ați pierdut telefonul într-un mall. Cum ați căuta logic în toate magazinele, astfel încât să nu ratați niciunul și să nu vă întoarceți în locuri deja verificate?”	Conversația
Reactualizarea cunoștințelor	**Întrebări scurte**: Ce este un graf? Ce sunt nodurile și muchiile? Cum reprezentăm un graf prin listă de adiacență?	Evaluare frontală
Informarea elevilor	**Titlul lecției** este scris pe tablă: Parcurgerea grafurilor orientate și neorientate în adâncime și în lățime	Conversația
Prezentarea materialului nou	**Noțiuni teoretice:** • **Parcurgerea grafurilor:** o BFS (Breadth-First Search) – Parcurgere în lățime. Se explorează nodurile apropiate mai întâi, apoi cele mai îndepărtate. Ideal pentru calculul distanțelor minime între noduri. Folosește o coadă și un vector de tip visited[]. Parcurgerea are loc pe „niveluri” – fiecare nod duce la vecinii săi, care la rândul lor duc la ai lor. o DFS (Depth-First Search) – Parcurgere în adâncime. Se explorează cât mai profund o ramură, înainte de a reveni. Este util pentru: detectarea ciclurilor, componente conexe, etc. Se poate implementa recursiv sau cu stivă. • **Exemple concrete afișate vizual:** o Se desenează un graf simplu pe tablă (6 noduri, 7 muchii). o Se indică pas cu pas ordinea de parcurgere cu BFS și DFS, cu săgeți și marcaje de vizitare. • **Aplicații reale ale parcurgerilor:**	Explicația, demonstrația, videoproiector, conversația

	○ BFS: Căutări în grafuri (ex: Google Maps), rețele sociale (niveluri de prietenie). ○ DFS: Detectare bucle în sisteme, generare labirinturi, compilatoare (analiză sintactică).	
Dirijarea învățării	Realizați implementarea parcurgerilor BFS și DFS în cadrul unui **graf neorientat**, utilizând limbajul C++. **Etape de lucru:** 1. Construiți graful folosind lista de adiacență. 2. Implementați algoritmul BFS cu următoarea structură: ○ Inițializați o coadă și un vector de vizite. ○ Adăugați nodul de start și marcați-l ca vizitat. ○ Parcurgeți fiecare vecin nevizitat și adăugați-l în coadă. ○ Afișați ordinea vizitării. 3. Implementați algoritmul DFS recursiv: ○ Marcați nodul curent ca vizitat. ○ Parcurgeți recursiv fiecare vecin nevizitat. ○ Afișați nodurile în ordinea parcurgerii. Realizați implementarea parcurgerilor BFS și DFS într-un **graf orientat**, utilizând limbajul C++. Observați diferențele față de grafurile neorientate. **Etape de lucru:** 1. Construiți graful folosind lista de adiacență. 2. Implementați algoritmul BFS pentru graf orientat: ○ Se folosește aceeași logică ca la graf neorientat, dar nu se adaugă muchii în ambele sensuri. ○ Parcurgerea respectă direcția muchiilor. 3. Implementați algoritmul DFS recursiv: ○ Apelați recursiv DFS doar pe vecinii direcți, fără întoarcere. ○ Exemplu de ieșire diferită față de graf neorientat. 4. Observați comportamentul: ○ Comparați rezultatele celor două parcurgeri.	Calculatorul, fișa de lucru, exercițiul, explicația
Asigurarea conexiunii inverse	Elevii testează algoritmii BFS și DFS pe **grafuri diferite**, inclusiv cu noduri izolate sau componente separate. Ce noduri nu au fost atinse? De ce? Care este diferența în ordinea de parcurgere între cele două metode?	Calculatorul, învățare prin descoperire
Asigurarea reținerii	Elevii completează **fișa de lucru** cu cerințe: implementarea BFS și DFS, observarea modificărilor în output în urma schimbării structurii grafului.	Fișa de lucru, explicații, Temă acasă

	Tema pentru acasă: Redactați un program care identifică toate componentele conexe dintr-un graf orientat.	
Obținerea de performanță	**Sarcină practică**: Graful unei rețele sociale – identificați prietenii de nivel 1 și 2 față de un utilizator. Ce algoritm folosiți?	Exercițiu aplicativ, lucrul în grupuri
Asigurarea transferului	**Discuție ghidată**: unde mai putem aplica BFS și DFS în viața reală (rețele, internet, jocuri)?	Conversație
Evaluare	Profesorul oferă **feedback verbal, corectează fișa de lucru.** Codurile implementate de elevi sunt verificate.	Evaluare orală și practică