LUCRAREA DE LABORATOR Nr. 3

TEMA: ALGORITMUL DE CĂUTARE ÎN ADÂNCIME ŞI LĂRGIME

1. SCOPUL LUCRĂRII:

- Studierea algoritmilor de căutare în graf şi a diferitor forme de păstrare şi prelucrare a datelor.
- Elaborarea procedurii de căutare în adâncime și lărgime.

➢ PREZENTAREA LUCRĂRII:

- Lucrarea va fi prezentată la lecția de laborator.
- Prezentarea cu întârziere se penalizează cu -1 pentru fiecare săptămână.

2. NOTE DE CURS

Structuri de date: ***liste***

Fiecare tip de listă defineşte o mulţime de şiruri finite de elemente de tipul declarat. Numărul de elemente care se numeşte lungimea listei poate varia pentru diferite liste de acelaşi tip. Lista care nu conţine nici un element se va numi vidă. Pentru listă sunt definite noţiunile începutul, sfârşitul listei şi respectiv primul şi ultimul element, de asemenea elementul curent ca şi predecesorul şi succesorul elementului curent. Element curent se numeşte acel unic element care este accesibil la momentul dat.

Structuri de date : ***fire de aşteptare***

Firele de aşteptare (*FA*, rând, coadă, şir de aşteptare) se vor folosi pentru a realiza algoritmul de prelucrare a elementelor listei în conformitate cu care elementele vor fi eliminate din listă în ordinea în care au fost incluse în ea (primul sosit - primul servit).

Operaţiile de bază cu firele de aşteptare:

- Formarea unui FA vid;
- Verificare dacă FA nu este vid;
- Alegerea primului element cu eliminarea lui din FA;
- Introducerea unei valori noi în calitate de ultim element al FA.

Structuri de date: ***stive***

Stiva se utilizează pentru a realiza algoritmul de prelucrare a elementelor după principiul "ultimul sosit - primul prelucrat" (LIFO).

Operaţiile de bază cu stivele sunt următoarele:

- Formarea unei stive vide;
- Verificare la vid;
- Alegerea elementului din topul stivei cu sau fără eliminare;
- Introducerea unui element nou în topul stivei.

Structuri de date - ***arbori***

Se va defini o mulţime de structuri fiecare din care va consta dintr-un obiect de bază numit *vârf* sau *rădăcina arborelui* dat şi o listă de elemente din mulţimea definită, care (elementele) se vor numi *subarbori* ai arborelui dat. Arborele pentru care lista subarborilor este vidă se va numi *arbore trivial*, iar rădăcina lui - *frunză*.

Rădăcina arborelui se va numi tatăl vârfurilor care servesc drept rădăcini pentru subarbori; aceste vârfuri se vor mai numi copiii rădăcinii arborelui: rădăcina primului subarbore se va numi *fiul cel mai mare,* iar rădăcina fiecărui subarbore următor în listă se va numi frate.

Operaţiile de bază pentru arbori vor fi:

- Formarea unui arbore trivial;
- Alegerea sau înlocuirea rădăcinii arborelui;
- Alegerea sau înlocuirea listei rădăcinilor subarborilor;
- Operaţiile de bază care sunt valabile pentru liste.

Căutare în adâncime

La căutarea în adâncime (parcurgerea unui graf în sens direct, în preordine) vârfurile grafului vor fi vizitate în conformitate cu următoarea procedură recursivă:

mai întâi va fi vizitată rădăcina arborelui q, apoi, dacă rădăcina arborelui nu este frunză - pentru fiecare fiu p al rădăcinii q ne vom adresa recursiv procedurii de parcurgere în adâncime pentru a vizita vârfurile tuturor subarborilor cu rădăcina p ordonate ca fii ai lui q.

În cazul utilizării unei stive pentru păstrarea drumului curent pe arbore, drum care începe din rădăcina arborelui şi se termină cu vârful vizitat în momentul dat, poate fi realizat un algoritm nerecursiv de forma:

```
Vizitează rădăcina arborelui şi introdu-o în stiva vidă S;
WHILE stiva S nu este vidă DO
        BEGIN
        fie p - vârful din topul stivei S;
        IF fiii vârfului p încă nu au fost vizitaţi
        THEN vizitează fiul mai mare al lui p şi introduce-l în S
                ELSE BEGIN
                        elimină vârful p din stiva S
                        IF p are fraţi THEN vizitează pe fratele lui p şi introduce-l în stiva S
                END
        END
```

Acest algoritm poate fi modificat pentru a putea fi utilizat la parcurgerea tuturor vârfurilor unui graf arbitrar. În algoritmul de mai jos se va presupune că este stabilită o relaţie de ordine pe mulţimea tuturor vârfurilor grafului, iar mulţimea vârfurilor adiacente cu un vârf arbitrar al grafului este de asemenea ordonată:

```
WHILE va exista cel puţin un vârf care nu a fost vizitat DO
        BEGIN
        fie p - primul din vârfurile nevizitate;
        vizitează vârful p şi introduce-l în stiva vidă S;
                WHILE stiva S nu este vidă DO
                        BEGIN
                        fie p - vârful din topul stivei S;
                        IF m vârfuri ale lui p sunt vârfuri adiacente nevizitate
                                THEN BEGIN
                                fie z primul vârf nevizitat din vârfurile adiacente cu p;
                                parcurge muchia (p,z), vizitează vârful z şi introduce-l în stiva S;
                                END
                        ELSE elimină vârful p din stiva S
                        END
        END
```

În cazul în care se va lucra cu un graf conex arbitrar cu relaţia de ordine lipsă, nu va mai avea importanţă ordinea de parcurgere a vârfurilor. Propunem un algoritm care utilizează mai larg posibilităţile stivei, cea ce face programul mai efectiv în sensul diminuării timpului de calcul necesar. De exemplu, acest algoritm în varianta recursivă este pe larg utilizat în programele de selectare globală în subdirectori (cazul programelor antivirus).

```
Introdu în stivă vârful iniţial şi marchează-l;
WHILE stiva nu este vidă DO
        BEGIN
        extrage un vârf din stivă;
        IF există vârfuri nemarcate adiacente cu vârful extras
                THEN marchează-le şi introduce-le în stivă;
        END
```

Algoritmul de căutare în lărgime

Parcurgerea grafului în lărgime, ca şi parcurgerea în adâncime, va garanta vizitarea fiecărui vârf al grafului exact o singură dată, însă principiul va fi altul. După vizitarea vârfului iniţial, de la care va începe căutarea în lărgime, vor fi vizitate toate vârfurile adiacente cu vârful dat, apoi toate vârfurile adiacente cu aceste ultime vârfuri ş.a.m.d. până vor fi vizitate toate vârfurile grafului. Evident, este necesar ca graful să fie conex. Această modalitate de parcurgere a grafului (în lărgime sau postordine), care mai este adesea numită parcurgere în ordine orizontală, realizează parcurgerea vârfurilor de la stânga la dreapta, nivel după nivel.

Algoritmul de mai jos realizează parcurgerea în lărgime cu ajutorul a două fire de aşteptare *O1* şi *O2*.

```
Se vor forma două fire de aşteptare vide O1 şi O2;

Introduce rădăcina în FA O1;

WHILE cel puţin unul din firele de aşteptare O1 sau O2 nu va fi vid DO

        IF O1 nu este vid THEN

                BEGIN
                    fie p vârful din topul FA O1;
                    vizitează vârful p eliminându-l din O1;
                    vizitează pe toţi fiii lui p în FA O2, începând cu cel mai mare;

                END

        ELSE
        în calitate de O1 se va lua FA O2, care nu este vid,
        iar în calitate de O2 se va lua FA vid O1;
```

Vom nota că procedura parcurgerii grafului în lărgime permite să realizăm arborele de căutare şi în acelaşi timp să construim acest arbore. Cu alte cuvinte, se va rezolva problema determinării unei rezolvări sub forma vectorului (a_1, a_2,...) de lungime necunoscută, dacă este cunoscut că există o rezolvare finită a problemei.

Algoritmul pentru cazul general este analogic cu cel pentru un graf în formă de arbore cu o mică modificare care constă în aceea că fiecare vârf vizitat va fi marcat pentru a exclude ciclarea algoritmului.

Algoritmul parcurgerii grafului în lărgime:

Se vor defini două FA O_1 şi O_2 vide;

Introdu vârful iniţial în FA O_1 şi marchează-l;

```
        WHILE FA O1 nu este vid DO
        BEGIN
                vizitează vârful din topul FA O1 şi elimină-l din FA;
                IF există vârfuri nemarcate adiacente cu vârful dat THEN introdu-le în FA O2;
        END
```

3. SARCINA DE BAZĂ

1. Elaboraţi procedura căutării în adâncime şi lărgime într-un graf arbitrar;
2. Elaboraţi un program cu următoarele posibilităţi:

- introducerea grafului în calculator,
- parcurgerea grafului în adâncime,
- vizualizarea rezultatelor.

4. ÎNTREBĂRI DE CONTROL

1. Definiţi structurile principale de date: liste, fire de aşteptare, stive, arbori.
2. Care sunt operaţiile definite pentru aceste structuri de date?
3. Care este principiul de organizare a prelucrării elementelor în firele de aşteptare şi în stive?
4. Definiţi noţiunea de parcurgere a grafului în adâncime/lărgime.
5. Ce fel de structuri de date se vor utiliza în căutarea în adâncime/lărgime?
6. Exemplificaţi utilizarea algoritmului de căutare în adâncime/lărgime.