

Structuri de date și algoritmi

(tema 2 - S1)

Exercitiul 1.

Descrieți o reprezentare pentru numere întregi care să nu aibă nicio restricție legată de valoarea maximă ce poate fi reprezentată (în afară de restricțiile legate de resursele de memorie ale sistemului de calcul). Descrieți pe scurt cum poate fi folosită această reprezentare pentru calcule de adunare, înmulțire și ridicare la putere.

O astfel de reprezentare ar putea fi realizată folosind o structură de date dinamică (un vector de exemplu) pentru a stoca cifrele numărului.

De exemplu, numărul 12345 poate fi stocat ca $\{1, 2, 3, 4, 5\}$. Un bit sau un element poate fi folosit pentru a stoca semnul numărului (pozitiv sau negativ).

Operațiile:

- Adunare: Se adună fiecare bloc corespunzător pentru două numere și se propaga transportul dacă este necesar, iar operația se continuă până la ultimul bloc.

$$\begin{array}{l} \{1, 2, 3\} \\ 3 + 6 = 9 \\ 2 + 5 = 7 \\ 1 + 4 = 5 \end{array} \quad \Rightarrow \quad \{5, 7, 9\}$$

- Înmulțire: Se înmulțește fiecare bloc din primul număr cu fiecare bloc din al doilea număr (ca metoda clasică de înmulțire pe hârtie). Din nou se propaga transportul și se continuă până la ultimul bloc în același fel.

$$\begin{array}{l} \{1, 2\} \\ 2 \cdot 4 = 8 \\ 1 \cdot 3 = 3 \end{array} \quad \Rightarrow \quad \{3, 8\}$$

- Ridicare la putere: Putem folosi algoritmul de exponențiere rapidă, adică se împart calculele în înmulțiri repetate. În cazul unui exponent par, putem înmulți două puteri identice (pentru $2^8 \rightarrow 2^4 * 2^4 \Leftrightarrow (2^2 * 2^2) * (2^2 * 2^2)$) și tot așa. În cazul unui exponent impar, se mai adaugă încă o înmulțire suplimentară.

Exercitiul 2

Ex2. Definiți un TDA listă de întregi. Specificați ce operații doriți să fie definite pentru acest TDA.

Apoi propuneți un fișier de tip header pentru o implementare a unei biblioteci în C pentru TDA abstract definit anterior.

```
typedef struct Lista{
    int *elemente;
    int dimensiune_curenta;
    int capacitate;
}Lista_t;
```