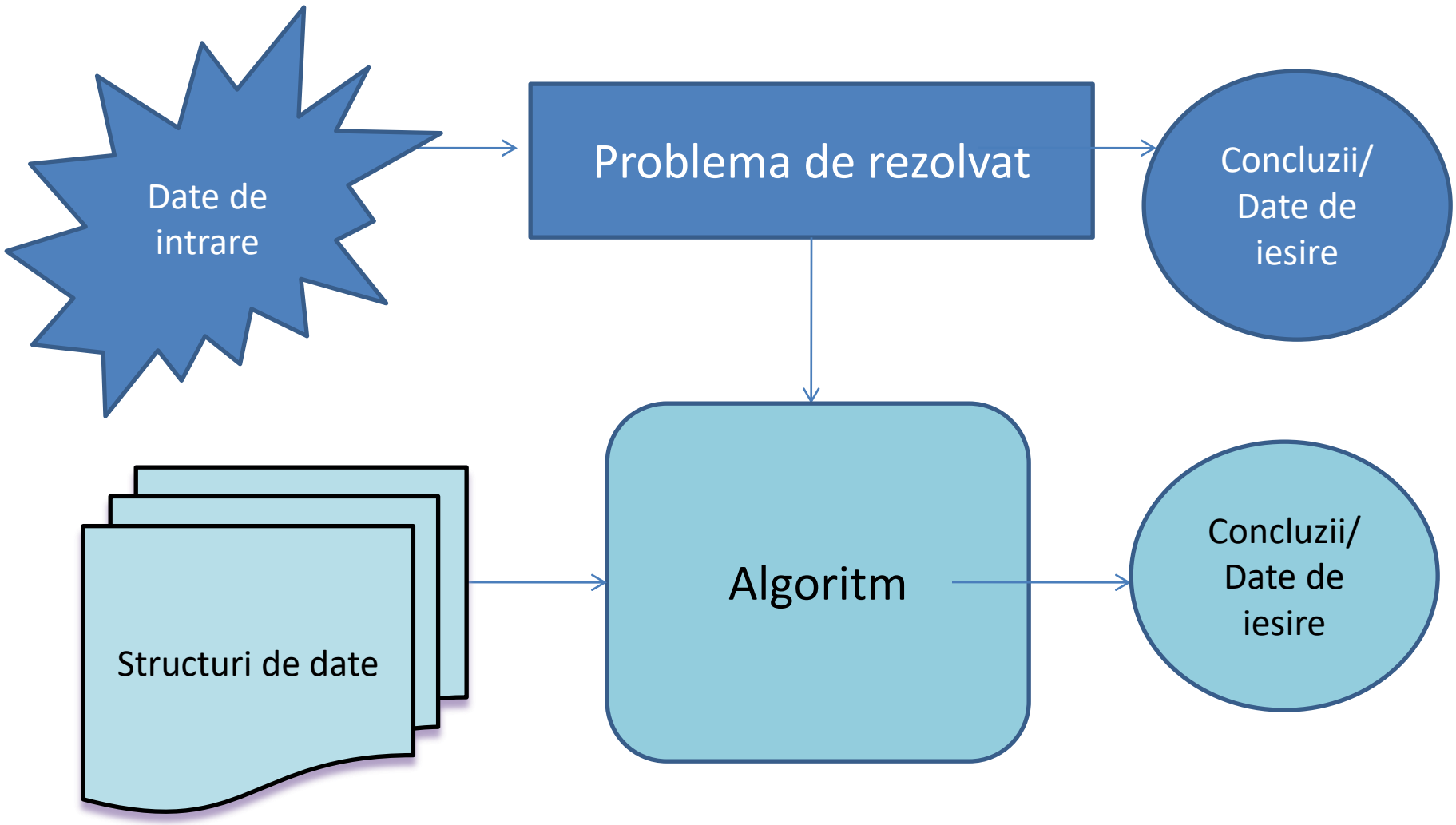
ALGORITMI SI STRUCTURI DE DATE

Curs 1

An univ 2022-2023


Date de intrare
Problema de rezolvat
Concluzii/ Date de iesire
Structuri de date
Algoritm
Concluzii/ Date de iesire

Structuri de date

- Structurile de date sunt colectii de date cu un anumit mod de organizare in asa fel incat sa fie eficiente pentru rezolvarea anumitor probleme.

- Structura de date = date + mod de organizare

Operatii

- Operatii de baza:
  - Inserarea unui elem
  - Stergerea unui elem
  - Accesarea unui elem

- Alte operatii:
  - Cautarea unui elem
  - Modificarea unui elem
  - Citirea/scrierea unui elem
  - Sortarea elem

Operatii

- Oricare din aceste operatii nu trebuie sa modifice structura colectiei de date (de ex, inserarea unui nou element intr-un graf sa genereze tot un graf, etc).
- Este important ca structura rezultata sa aiba aceleasi proprietati dupa executarea operatiei.

Structuri de date liniare

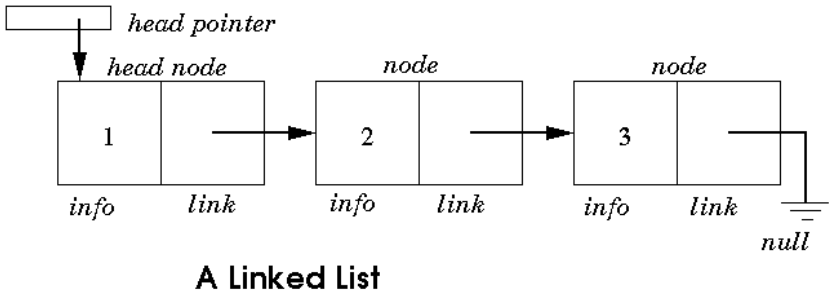
- O structura de date liniara = listă = o colectie finita de elemente de acelasi tip, ordonate liniar, adică există un element considerat primul in listă, un element considerat ultimul și pentru orice alt element din listă există un element precedent și un element următor.

Reprezentari liste

- Alocare secventiala
- In alocarea secventiala, elementele listei sunt memorate in locatii consecutive de memorie.
- Ati mai intalnit, in programare, liste alocate secvential ?

  Desigur. Vectorii.

- Alocare inlantuita



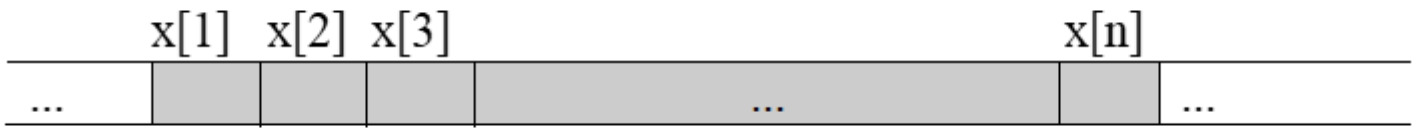
A Linked List

http://people.engr.ncsu.edu/efg/210/s99/Notes/LinkedList.1.html

Liste in alocare secventiala

- Toate elementele se identifica printr-un index care precizeaza pozitia elementului in lista:
- primul element va avea indicele 1
- al doilea element va avea indicele 2
- etc.
- Lista este identificata prin numele colectiei ei. Notam cu x numele listei.
- Notam n= nr de elemente din lista.
- Notam N= numarul maxim posibil de elemente ale sirului.

Liste in alocare secventiala



Lista vida echivalent cu n=0

Lista plina echivalent cu n=N

Operatiile cu liste alocate secvential

- **accesarea/ modificarea unui element**
  - se face prin intermediul indicelui elementului
- **inserarea unui element**
  - se poate face intr-o pozitie data, dupa sau inaintea unui element dat: inserarea unui element numit *elem_nou* intr-o pozitie data *k*
- **stergerea unui element**:
  - a elementului din pozitia data *k*

Accesarea/ modificarea unui element

- Accesarea elementului al k-lea. $1 \leq k \leq n.$

x[k], P $1 \leq k \leq n.$

- Modificare elementului al k-lea. $1 \leq k \leq n.$

x[k]=elem_nou

Inserarea unui element

- inserarea unui element numit *elem_nou* intr-o pozitie data *k*
- Presupunem $1 \leq k \leq n+1$.
- Daca n = N atunci se produce OVERFLOW adica spatiul alocat listei este ocupat in totalitate si ca nici o alta inserare nu este posibila.
- In caz contrar, se muta elementele sirului x[k], ..., x[n], cate un bloc de memorie spre dreapta, incepand cu ultimul.
- Se introduce elementul nou in pozitia k.
- Se actualizeaza numarul de elemente al sirului.

inserarea unui element numit *elem_nou* intr-o pozitie data *k*

```
if n = N then OVERFLOW
          else for i = n, k, -1
                              x[i+1] = x[i]
                    endfor
endif
x[k] = elem_nou
n = n +1
```

Stergerea unui element

- stergerea elementului din pozitia data *k*

- Daca n = 0 atunci se produce UNDERFLOW adica lista este vida si deci eliminarea unui element din lista nu este posibila.
- Presupunem $1 \leq k \leq n$. Se salveaza informatia continuta de elementul de indice k pentru cazul in care se doreste prelucrarea ei.
- Se muta elementele sirului x[k+1], ..., x[n], cate un bloc de memorie spre stanga, incepand cu x[k+1].
- Se actualizeaza numarul de elemente al sirului.

stergerea elementului din pozitia data *k*

```
if n = 0 then UNDERFLOW
        else elem_sters=x[k]
             for i = k, n-1, -1
                 x[i] = x[i+1]
             endfor
endif
n = n-1
```