STRUCTURI DE DATE

Lista simpla Lista dubla

LISTA LINIARA

Lista liniara:

- Colectie de elemente denumite *noduri*;
- Relatie de ordine rezultata din pozitia nodurilor;
- Elemente de acelasi tip structura omogena;
- Lungimea listei numarul de noduri;
- Asemanatoare structurii de tip masiv unidimensional;
- Structura alocata in heap gestionata prin variabila pointer.

http://www.acs.ase.ro

https://github.com/mpopaeu/struct

LISTA LINIARA

Lista liniara vs. Vector:

Lista liniara	Vector
Numar variabil de elemente, fara declararea anticipata a dimensiunii	Numar predeterminat de elemente, cu rezervarea de memorie pe dimensiunea declarata
Elemente alocate la adrese nu neaparat consecutive	Spatiu contiguu de memorie (elementele sunt adiacente logic si fizic)
Alocare memorie la executie	Alocare memorie la compilare/executie
Referire secventiala a nodurilor pornind de la adresa primului nod	Referire prin deplasament al elementului fata de adresa de inceput a zonei

http://www.acs.ase.ro

https://github.com/mpopaeu/structuri



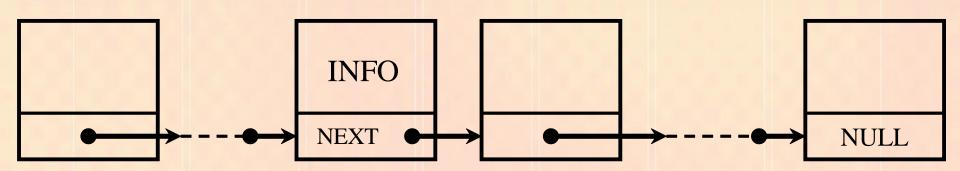
LISTA LINIARA

Implementare:

- Utilizare variabile pointer;
- Nodurile contin doua categorii de informatii:
- Campuri cu informatia structurala;
- Campuri cu informatia de legatura adrese ale altor noduri din cadrul listei

Lista simpla:

- Lista liniara;
- Un singur camp cu informatia de legatura nodul succesor al campului curent;
- Ultimul nod nu are succesor informatia de legatura este nula (NULL sau 0);
- Gestionata prin variabila pointer cu adresa primului nod din lista liniara.



Definirea structurii unui nod dintr-o lista simpla:

```
struct Nod {
    char inf; // date de prelucrat
    Nod* next; // date de legatura
};
```

http://www.acs.ase.ro

6

https://github.com/mpopaeu/structuri

Declararea unei liste simple vide:

```
Nod *prim = NULL;
```

Alocarea de memorie heap pentru un nod al liste simple:

```
Nod *nou = new Nod;
```

Dezalocarea de memorie heap pentru un nod al liste simple:

delete nou;

http://www.acs.ase.ro

https://github.com/mpopaeu/structur

Initializarea si referirea informatiei utile dintrun nod al listei simple:

```
nou->inf = 'A';
(*nou).inf
```

Initializarea si referirea informatiei de legatura dintr-un nod al listei simple:

```
nou->next = NULL;
```

Operatii cu liste simple:

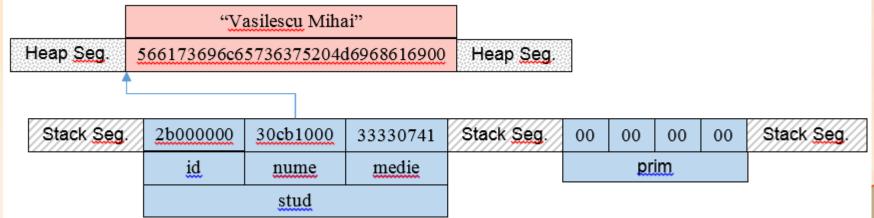
- Creare listei;
- Inserare nod;
- Traversare;
- Cautare;
- Interschimbare noduri;
- Stergere nod;
- Stergerea listei;
- Sortarea listei;
- Concatenarea de liste;
- Interclasare liste.

http://www.acs.ase.ro

https://github.com/mpopaeu/structur



Creare listei



http://www.acs.ase.ro https://github.com/mpopaeu/structuri

10

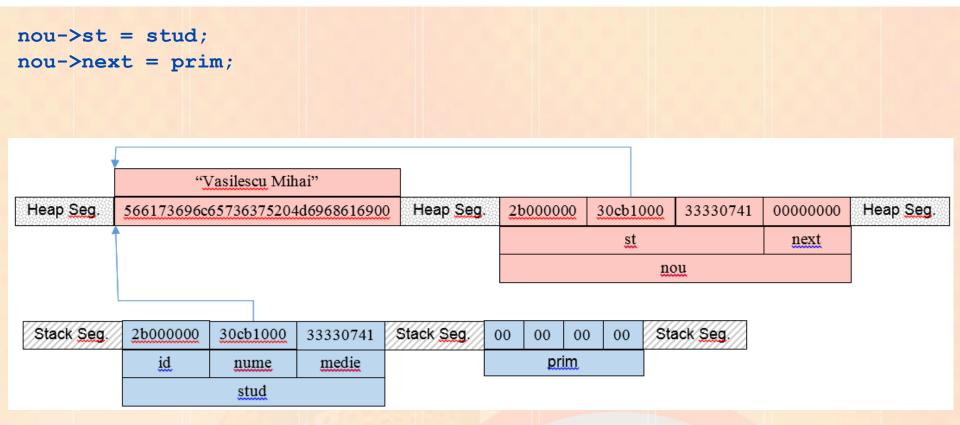
```
Nod* nou;
nou = (Nod*)malloc(1 * sizeof(Nod));
                      "Vasilescu Mihai"
 Heap Seq.
                                                             cdcdcdcd
                                                                        cdcdcdcd
                                                                                             cdcdcdcd
                                                                                                        Heap Seg.
             566173696c65736375204d6968616900
                                                 Heap Seq.
                                                                                   cdcdcdcd
                                                                           <u>st</u>
                                                                                               next
                                                                               nou
                                                                               Stack Seg.
  Stack Seg.
              2b000000
                         30cb1000
                                    33330741
                                               Stack Seq.
                                                           00
                                                                00
                                                                    00
                                                                         00
                 id
                                     medie
                                                                 prim
                           nume
                           stud
```

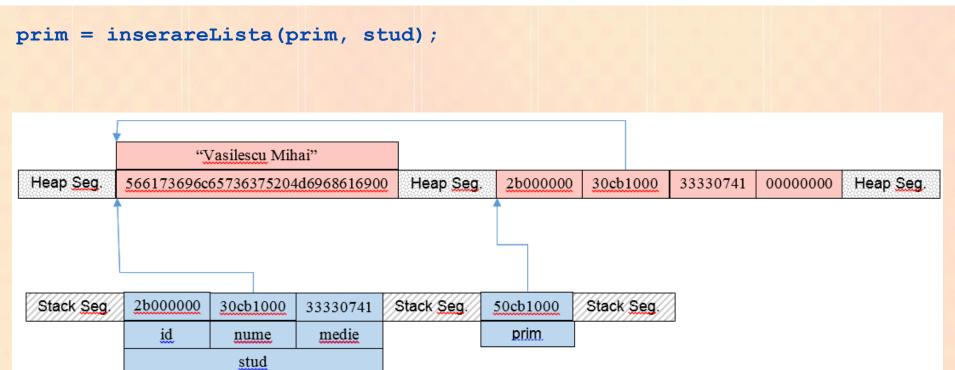
http://www.acs.ase.ro

https://github.com/mpopaeu/structuri

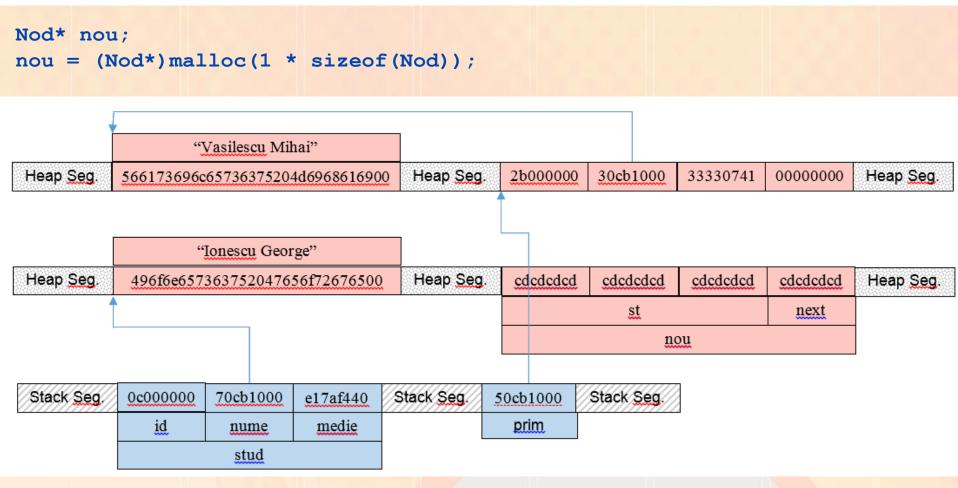
11

110101010101110<mark>10000</mark>

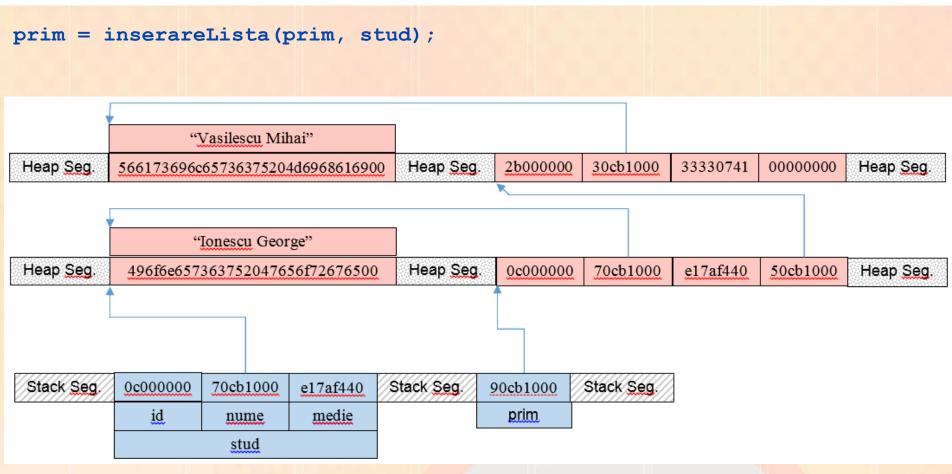




```
stud.id = atoi(token); // 12
stud.nume = (char*)malloc((strlen(token) + 1) * sizeof(char));
strcpy(stud.nume, token); // Ionescu George
stud.medie = atof(token); // 7.64
                  "Vasilescu Mihai"
 Heap Seg.
           566173696c65736375204d6968616900
                                         Heap Seq.
                                                   2b000000
                                                            30cb1000
                                                                      33330741
                                                                               00000000
                                                                                        Heap Seg.
                   "Ionescu George"
 Heap Seg.
                                         Heap Seg.
            496f6e657363752047656f72676500
 Stack Seq.
           0c000000
                     70cb1000
                                       Stack Seg.
                                                           Stack Seq.
                              e17af440
                                                  50cb1000
                               medie
                                                    prim
              id
                      nume
                       stud
```



```
nou->st = stud;
nou->next = prim;
                      "Vasilescu Mihai"
 Heap Seg.
                                                             2b000000
                                                                                   33330741
                                                                                                         Heap Seq.
             566173696c65736375204d6968616900
                                                 Heap Seg.
                                                                        30cb1000
                                                                                              00000000
                      "Ionescu George"
 Heap Seq.
                                                 Heap Seq.
                                                             0c000000
                                                                        70cb1000
                                                                                   e17af440
                                                                                              50cb1000
                                                                                                          Heap Seg.
              496f6e657363752047656f72676500
                                                                            st
                                                                                                next
                                                                                nou
 Stack Seg
              0c000000
                         70cb1000
                                    e17af440
                                               Stack Seq.
                                                           50cb1000
                                                                       Stack Seq.
                                     medie
                                                              prim
                 id
                           nume
                           stud
```



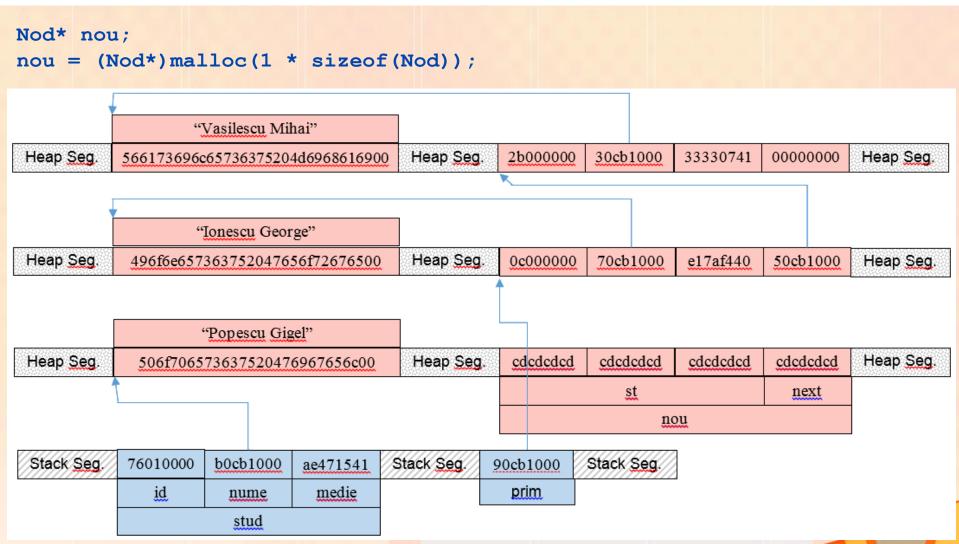


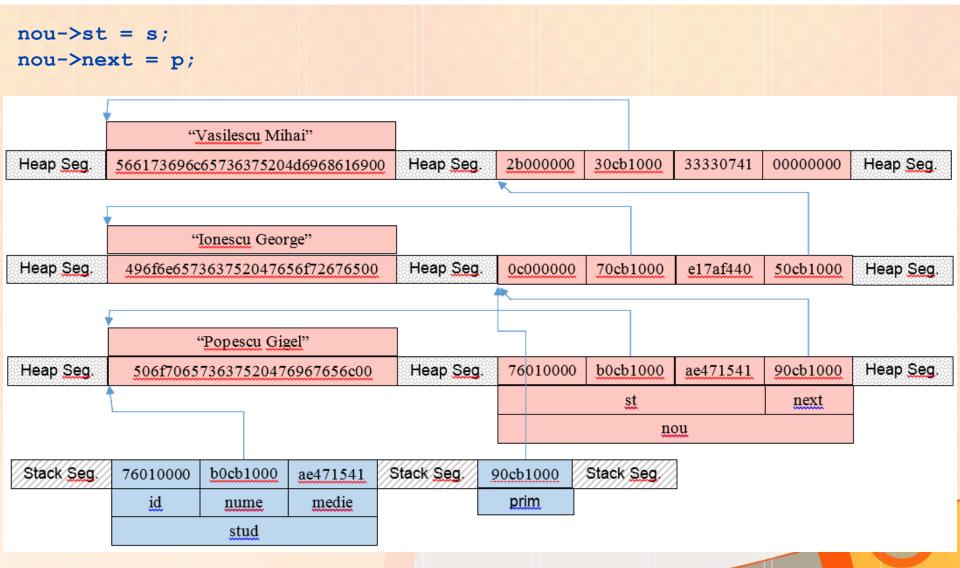
```
stud.id = atoi(token); // 374
stud.nume = (char*)malloc((strlen(token) + 1) * sizeof(char));
strcpy(stud.nume, token); // Popescu Gigel
stud.medie = atof(token); // 9.33
                   "Vasilescu Mihai"
 Heap Seg.
                                          Heap Seq.
                                                     2b000000
                                                              30cb1000
                                                                        33330741
                                                                                 00000000
                                                                                           Heap Seg.
           566173696c65736375204d6968616900
                   "Ionescu George"
 Heap Seq.
                                          Heap Seg.
            496f6e657363752047656f72676500
                                                     0c000000
                                                              70cb1000
                                                                                  50cb1000
                                                                                           Heap Seq.
                                                                        e17af440
                    "Popescu Gigel"
 Heap Seg
                                           Heap Seq.
             506f706573637520476967656c00
 Stack Seg
            76010000
                     b0cb1000
                               ae471541
                                        Stack Seq.
                                                   90cb1000
                                                             Stack Seq.
               id
                                medie
                                                     prim
                       nume
                       stud
```

http://www.acs.ase.ro

110101010101111010000

https://github.com/mpopaeu/structuri





110101010101111010000

prim = inserareLista(prim, stud); "Vasilescu Mihai" Heap Seg. Heap Seq. 2b000000 30cb1000 33330741 00000000 Heap Seg. 566173696c65736375204d6968616900 "Ionescu George" Heap Seq. Heap Seg. 0c000000 70cb1000 e17af440 50cb1000 Heap Seq. 496f6e657363752047656f72676500 "Popescu Gigel" Heap Seq. Heap Seg. Heap Seq. 76010000 b0cb1000 ae471541 90cb1000 506f706573637520476967656c00 Stack Seg 76010000 b0cb1000 ae471541 Stack Seq. d0cb1000 Stack Seq prim id nume medie stud

Lista dubla:

- Lista liniara;
- Doua campuri cu informatii de legatura nodurile succesor, respectiv predecesor ale campului curent;
- Ultimul nod nu are succesor informatia de legatura este nula (NULL sau 0);
- Primul nod nu are predecesor informatia de legatura este nula (NULL sau 0);

http://www.acs.ase.ro

22

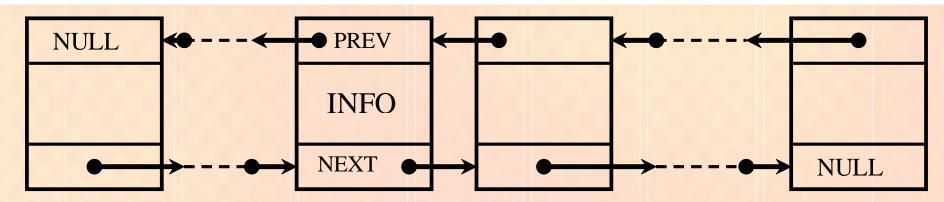
Lista dubla (continuare):

- Gestionata prin variabile pointer: una retine adresa primului nod din lista liniara, iar cealalta retine adresa ultimului nod din lista liniara;
- Posibilitatea gestionarii structurii printr-o structura articol ce incapsuleaza cei doi pointeri.

http://www.acs.ase.ro

23

https://github.com/mpopaeu/structur



Definirea structurii unui nod dintr-o lista dubla:

```
struct NodD{
    char inf;
    NodD *prev, *next;
};
```

Incapsularea adreselor primului si ultimului nod ale unei liste duble vide:

```
struct ListaD{
    NodD *prim, *ultim;
};
```

Declararea unei liste duble vide:

```
ListaD lst;
lst.prim = NULL;
lst.ultim = NULL;
```

Alocarea de memorie heap pentru un nod al liste duble:

```
NodD *Nou = new NodD;
```

25

http://www.acs.ase.ro

https://github.com/mpopaeu/structur

Dezalocarea de memorie heap pentru un nod al liste duble:

```
delete Nou;
```

Initializarea si referirea informatiei utile dintrun nod al listei duble:

```
Nou->inf = 'Z';
```

Initializarea si referirea informatiilor de legatura dintr-un nod al listei duble:

```
Nou->next = NULL;
Nou->prev = NULL;
```

Referirea informatiilor din primul nod al listei duble:

```
lst.prim->inf;
lst.prim->next;
lst.prim->prev;
```

Referirea informatiilor din ultimul nod al listei duble:

```
lst.ultim->inf;
lst.ultim->next;
lst.ultim->prev;
```

Operatii cu liste duble – similare operatiilor cu liste simple