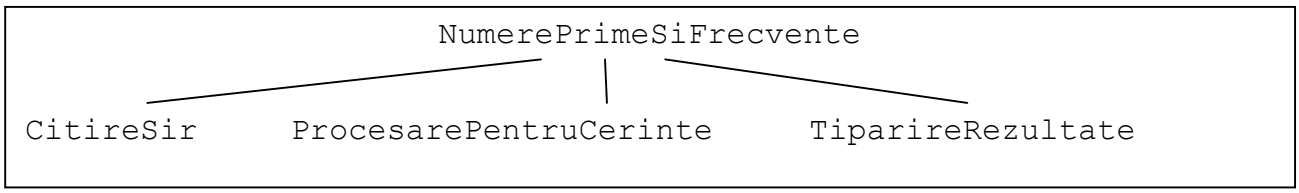
Proiectare functionalitati (si subfunctionalitati)

- **Alegerea reprezentarii pentru tipul sir si pentru memorarea/retinerea cerintelor.**
  - ds – dimensiunea sirului initial
  - vs – valorile sirului – sir/vector cu ds elemente
  - structura *nr_frecv*(numar, frecventa) pentru memorarea unui numar si a frecventei de aparitie in sirul initial;
  - dpf – dimensiunea sirului de rezultate;
  - vpf – valorile sirului de numere prime si a frecventei de aparitie a lor in sirul initial (tipul elementelor din sir va avea tipul *nr_frecv*.
- **Diagrama de structura pentru rezolvarea problemei**
  - Diagrama de structura (cu functionalitatile principale)



  - Algoritmul *NumerePrimeSiFrecvente*(ds,vs,dpf,vpf) este:

```
CitireSir(ds,vs);
ProcesarePentruCerinte(ds,vs,dpf,vpf);
TiparireRezultate(dpf,vpf);
SfarsitNumerePrimesiFrecvente;
```

- **Specificarea (date+rezultate+preconditii+postconditii) pentru fiecare functionalitate.**
  - Subalgoritmul CitireSir(ds,vs)
    - Date:- Precond:-true
    - Rezultate: ds,vs Postcond: ds – numar natural; vs[i] – numar natural, i=1,ds.

    unde:
    - ds - reprezinta dimensiunea sirului;
    - vs – reprezinta valorile sirului;
  - Subalgoritmul ProcesarePentruCerinte(ds,vs,dpf,vpf)
    - Date:ds,vs Precond:ds – nr natural; vs[i]–nr natural, i=1,ds
    - Rezultate: dpf,vpf Postcond: dpf – nr natural; vpf[i] – nr prim din sirul vs (campul numar), vpf[i] – frecventa de aparitie a nr vp[i] (campul frecventa) in sirul vs, i=1,dpf; vpf[i] – nr naturale.

    unde:
    - ds - reprezinta dimensiunea sirului initial;
    - vs – reprezinta valorile sirului initial;
    - dpf – reprezinta dimensiunea sirului de numere prime si de frecvente;
    - vpf – reprezinta numerele prime determinate din sirul initial (campul numar) si frecventele numerelor prime in sirul initial vs (campul frecventa).
  - Subalgoritmul TiparireRezultate(dpf,vpf)
    - Date:dpf, vpf Precond:-true
    - Rezultate: ds,vs Postcond: ds – numar natural; vs[i] – numar natural, i=1,ds.

    unde:

- dpf – reprezinta dimensiunea sirului de numere prime si de frecvente;
- vpf – reprezinta un sir de numere prime (campul numar) si un numar asociat fiecarui numar (campul frecventa).

- **Descrierea functionalitatilor (subalgoritmilor) in limbajul Pseudocod**
  - Subalgoritmul ProcesarePentruCerinte (ds,vs,dpf,vpf) este:

```
@Cat timp (mai sunt elemente de parcurs din sirul initial) exe.
  @Daca elementul curent din sirul vs este prim atunci
     @Daca (elementul curent nu se regaseste in vpf) atunci
        @Se adauga elementul in vpf cu frecventa 1
     @altfel
        @Se cauta pozitia in sirul vpf pe care se alfa
            Elementul current;
        @Se incrementeaza frecventa elementului curent
            in sirul vpf.
      @sfDaca
   @sfDaca
   @deplasarea la urmatorul element din sirul initial;
@SfCatTimp
```

Din descrierea algoritmului de mai sus se poate observa necesitatea a trei subalgoritmi:
- primul - care sa determine daca un numar se regaseste sau nu intr-un sir de elemente.
- Al doilea - care sa determine pozitia unui element intr-un sir de elemente.
- Al treilea – care sa verifice daca un numar este prim sau nu.

Primul si al doilea algoritm realizeaza fiecare o parcurgere a aceluiasi sir. Cei doi subalgoritmi pot fi redusi la unul singur care sa determine pozitia pe care se regaseste un element in sirul de elemente (pozitie valida intre 1 si dimensiunea sirului) sau daca nu se regaseste rezultatul sa fie -1. Astfel, descrierea subalgoritmului *ProcesarePentruCerinte* poate fi descrisa:

```
Subalgoritmul ProcesarePentruCerinte(ds,vs,dpf,vpf) este:
     contor ← 1;
     Cattimp (contor<=ds) executa
       Daca EstePrim(vs[contor]) atunci
         poz←CautaPozitie(vs[contor], dpf, vpf);
         Daca( poz= -1) atunci
             dpf ← dpf+1;
             vpf[dpf].numar ←vs[contor];
             vpf[dpf].frecventa←1;
         altfel
             vpf[poz].frecventa←vpf[poz].frecventa+1;
         sfDaca;
       sfDaca;
       contor←contor+1;
     SfCatTimp
SfProcesarePentruCerinte
```

Astfel, subalgoritmii (subfunctionalitatile) noi determinate a fi necesari sunt:
-F4 - *EstePrim (nr)* – verifica daca numarul *nr* este prim sau nu;
- *F5 - CautaPozitie(e,d,s)* – cauta pozitia pe care s-ar putea afla elemental *e* din sirul de dimensiune *d* cu elementele memorate in *s.*

- **Specificarea si descrierea subfunctionalitatilor (subalgoritmilor) in limbajul Pseudocod – determinati la pasul anterior**
  - Specificare *EstePrim* (nr)
    - Date:nr
      - Precond: nr –numar natural;

- Rezultate: true/fals
  - Postcond: (true si *nr* numar prim)sau(fals si *nr* nu e numar prim)

- Functia EstePrim(nr) este:

```
prim←true;
Daca (nr=0) sau (nr=1)  atunci prim←fals
altfel
  d←2;
  Cat timp (d<=nr div 2) and (not prim) executa
       Daca (nr mod d=0) atunci
            prim←fals
       altfel
            d←d+1;
       SfDaca;
  SfCatTimp
EstePrim←prim;
sfEstePrim
```

- Specificare *CautaPozitie* (e,d,s)
  - Date:e, d, s
    - Precond: e, d –numar natural; s – sir de elemente cu structura (numar,frecventa); d- dimensiunea sirului;e – elementul cautat;
  - Rezultate: pozitie
    - Postcond: (pozitie =-1 si e nu exista in sirul s) sau (1<=pozitie<=d si e exista in sir)

- Functia *CautaPozitie* (e, d, s) este:

```
poz←-1;
p←1;
Cat timp (p<=d) and (poz=-1) executa
       Daca (e=s[p]) atunci
            poz←p
       altfel
            p←p+1;
       SfDaca;
  SfCatTimp
  CautaPozitie←poz;
sfEstePrim
```

In urma specificarilor si a descrierilor in limbajul Pseudocod a subfunctionalitatilor noi adaugate se observa nu mai este nevoie de introducerea unor noi (sub)functionalitati.

Au fost determinate astfel, la faza de proiectare a tuturor functionalitatilor ce trebuie realizate.