TAD Rational

1) Etapa de specificare
   a. Specificarea domeniului
   b. Specificarea operatiilor TAD Rational
2) Etapa de proiectare
   a. Alegerea reprezentarii
   b. Specificarea (in limbajul Pseudocod) a operatiilor TAD conform reprezentarii alese.
3) Etapa de implementare
   a. Implementarea modulara a TAD Rational – Unit – in limbajul Pascal

---

1) Etapa de specificare

a. Specificarea domeniului

$$DRat = \left\{ q = \frac{numarator}{numitor} \mid numarator \in Z, numitor > 0, cmmdc(numarator, numitor) = 1 \right\}.$$

b. Specificarea operatiilor TAD Rational

Observatii:

1) Specificarea unui TAD este independenta de reprezentarea datelor si de implementarea operatiilor.
2) Trebuie create operatii suficiente pentru gestionarea/lucrul cu TAD.
3) Pentru specificarea fiecarei operatii a TAD-ului se vor preciza: Date+Preconditii si Rezultate+Postconditii.
4) Sugestie: numele fiecarei operatii sa fie precedate de denumirea TAD-ului.

Operatia **RationalCreare**(na,ni,r,corect)
Date: na,ni
Rezultate: r, corect

φ(X)::na,ni$\in Z$
$\psi(X, Z)$ ::
corect={T,F} si (((corect=T)si (r$\in DRat, numaratorul\ r = na, numitorul\ r = ni$))$sau$ $((corect = F) si (r\ nu\ este\ creat))$).

Operatia **RationalSelNumarator**(r,na);
Date: r
Rezultate:na

φ(X):: r$\in DRat$
$\psi(X, Z)$ :: na va contine numaratorul numarului r.

Operatia **RationalSelNumitor**(r,ni);
Date: r
Rezultate:ni

φ(X):: r$\in DRat$
$\psi(X, Z)$ :: ni va contine numitorul numarului r.

Andreea Vescan
avescan@cs.ubbcluj.ro

Operatia **RationalSetNumitor**(ni, r,corect);
Date:ni, r
Rezultate: r, corect
φ(X):: r∈ $DRat$, ni∈ Z si $ni \neq 0$
$\psi(X, Z)$ ::corect={T, F} si r∈ $DRat$ si
((corect=T si r are numitorul ni si ni∈ Z)
sau(corect=F si ni nu are numitorul din r si (ni=0 sau ni∉ Z)))

Operatia **RationalSetNumarator**(na, r, corect)
Date:na, r
Rezultate: r, corect
φ(X):: r∈ $DRat$, na∈ Z
$\psi(X, Z)$ :: corect={T, F} si r∈ $DRat$ si
((corect=T si r are numaratorul na si na∈ Z)
sau(corect=F si na nu are numitorul din r si na∉ Z))

Operatia **RationalLaIntreg**(r, n, corect)
Date: r
Rezultate: n, corect
φ(X):: r∈ $DRat$
$\psi(X, Z)$ :: corect={T, F} si (corect= T) si
N∈ Z si n retine numarul rational r convertit la intreg si
numitorul numarului rational r este 1) sau
(corect=F si numitorul numarului rational r nu este 1 si n∈ Z si
n nu retine numaratorul numarului rational r (eventual retine 0 ca si initializare))

Operatia **RationalCompara**(r1,r2, rez);
Date: r1, r2
Rezultate: rez
φ(X):: r1, r2∈ $DRat$
$\psi(X, Z)$ :: $rez = \{-1,0,1\} si$
$((rez = -1\ si\ r1 < r2) sau$
$(rez = 0\ si\ r1 = r2) sau$
$(rez = 1\ si\ r1 > r2))$

Operatia **RationalAdunare**(r1,r2,rs);
Date: r1, r2
Rezultate: rs
φ(X):: r1, r2∈ $DRat$
$\psi(X, Z)$ ::rs∈ $DRat\ si\ rs = r1 + r2$

Operatia **RationalLaString**(r, s);
Date: r
Rezultate: s
φ(X):: r∈ $DRat$
$\psi(X, Z)$ :: $s -$ sir de caractere si
s contine numaratorul si numitorul numarului rational r.

Operatia **RationalDinString**(s, r, corect);
Date: s
Rezultate: r, corect
φ(X):: $s -$ sir de caractere
$\psi(X, Z)$ :: $corect = \{T, F\} si$
$((corect = T\ si\ r\ \in DRat\ cu\ valori\ pentru\ numarator\ si\ numitor\ din\ stringul\ s)$
$sau(corect = F\ si\ conversia\ din\ stringul\ s\ la\ doua\ numere\ intregi\ nu\ s - a\ realizat\ cu\ succes))$

2) Etapa de proiectare

a. Alegerea reprezentarii

Reprezentarea posibila 1) – tip de date structurat cu doua campuri (atribute): numarator si numitor.

Andreea Vescan
avescan@cs.ubbcluj.ro

Reprezentarea posibila 2) – tablou unidimensional cu doua valori: prima valoare pentru numarator, a doua valoare pentru numitor.

Alegem reprezentarea 1). La pasul urmator (b), specificare operatiilor se va face in functie de reprezentarea aleasa.

Tipul TRational = (numa,numi);

b. Specificarea (in limbajul Pseudocod) a operatiilor TAD conform reprezentarii alese.

```
Functia RationalCreare(na,ni,r) este:
      corect ← true;
      Daca (ni=0) atunci corect  ←false
      altfel
         RationalCreareOk(na,ni,r);
      SfDaca;
      RationalCreare ← corect;
Sf RationalCreare;
```

Observatie: S-a adaugat metoda **RationalCreareOk** care va crea un rational r din na si ni daca ni nu este 0. Aceasta metoda nu va fi accesibila din exterior: este o metoda ”ajutatoare” implementarii TAD-ului.

```
Subalgoritmul RationalCreareOk(na,ni, r) este:
   r.numa ←na;
   r.numi ←ni;
   Daca (ni<0) atunci
         r.numa ←-r.numa;
         sfDaca
   Simplificare(r);
SfRationalCreareOk;
```

Observatie: S-a adaugat **Simplificare** care simplifica numarul rational (vezi definitia domeniului DRat). Aceasta metoda nu va fi accesibila din exterior: este o metoda ”ajutatoare” implementarii TAD-ului.

```
Subalgoritmul  Simplificare(r) este:
  Fie x ←r.numa;
  Fie y ←r.numi;

  Cattimp(x<>y) executa
     Daca(x>y)atunci
              X ←x-y
    altfel
              y←y-x;
    r.numa←r.numa div x;
    r.numi ←r.numi div x;
SfSimplificare;
```

```
Functia RationalSelNumarator(r) este:
   RationalSelNumarator ←r.numa;
Sf RationalSelNumarator
```


Andreea Vescan
avescan@cs.ubbcluj.ro

```
Functia RationalSelNumitor(r)este:
  RationalSelNumitor← r.numi;
sfRationalSelNumitor;

Functia RationalSetNumitor(ni, r) este:
   Fie corect ← true;
   Daca(ni=0) atunci corect←false
   altfel
        r.numi←ni;
        Daca (r.numi<0) atunci r.numa←-r.numa;
        Simplificare(r);
   SfDaca;
   RationalSetNumitor←corect;
SfRationalSetNumitor;

Functia RationalSetNumarator(na, r) este:
   Fie corect←true;
   r.numa←na;
   Simplificare(r);
   RationalSetNumarator←corect;
end;


Function RationalLaIntreg(r, n) este:
    Corect←true;
    Daca(r.numi<>1)atunci corect←false
    altfel
      n←r.numa;
SfRationalLaIntreg;

Functia RationalCompara(r1,r2) este:

   Cheama RationalCreareOk(r1.numa*r2.numi,r1.numi*r2.numi,r1Nou);
   Cheama RationalCreareOk(r2.numa*r1.numi,r1.numi*r2.numi,r2Nou);
   Daca(r1Nou.numa<r2Nou.numa) atunci
      rez← -1
   altfel
       Daca(r1Nou.numa=r2Nou.numa) atunci
           rez←0
       altfel
          Daca(r1Nou.numa>r2Nou.numa) atunci
            rez ←1;
 RationalCompara←rez;
SfRationalCompara;

Subalgoritmul  RationalAdunare(r1,r2,rs) este:
   rs.numa←r1.numa*r2.numi+r1.numi*r2.numa;
   rs.numi←r1.numi*r2.numi;
   Simplificare(rs);
SfRationalAdunare;
```


Andreea Vescan
avescan@cs.ubbcluj.ro

```
Subalgoritmul RationalLaString(r,s) este:
   Fie numa='', numi=''.
   str(r.numa,numa); //converteste din numar la string
   str(r.numi,numi); //converteste din numar la string
   s:=numa+'/'+numi; //concateneaza doua stringul in stringul s.
SfRationalLaString;

Functia RationalDinString(s, r) este:
 p←1;
 numa←0;
 corect←true;
 Cattimp(s[p]<>' ') si(p<=NumarulDeElementeDin_s) si (corect) executa
       val(s[p],cifra,codDeEroare); //transforma caracterul s[p] in numar memorat in variabila cifra
                                    // daca transformarea este corecta atunci variabila codDeEroare retine 0.
       Daca(codDeEroare>0) atunci
          corect←false
       altfel
          numa←numa*10+cifra;
          p←p+1;
      SfDaca;
 SfCattimp;
 numi←0;
 p←p+1;
 Cattimp(s[p]<>' ')and p<=NumarulDeElementeDin_s) si (corect) executa
   val(s[p],cifra,codDeEroare);
   Daca(codDeEroare>0) atunci
    Corect←false
   altfel
      numi←numi*10+cifra;
      p←p+1;
    SfDaca;
 SfCattimp;
 Daca (corect) atunci
   corect←RationalCreare(numa,numi,r);
 SfDaca;
 RationalDinString←corect;
SfRationalDinString;
```

3) Etapa de implementare

   a. **Implementarea modulara a TAD Rational – Unit – in limbajul Pascal**
   b. **Cod sursa in arhiva.**

Andreea Vescan
avescan@cs.ubbcluj.ro