Capitolul 4: 36 de exemple de programe în Rodin explicate

1. Structura unui program care *scrie* o valoare a unei constante:

```
{scrie 2010
}
```

Un program foarte, foarte simplu.

Avem un program care pur şi simplu afişează numărul 2010. Acest număr poate fi înlocuit cu un alt număr.

2. Operația de *I / O (INPUT / OUTPUT)*:

```
{text "Salutare Lume!";
 text "Cum va simtiti astazi?"
}
```

Acest program conține doar afişarea unor mesaje (adică a unor date de tip string)

3. Citirea, procesarea, afişarea:

```
{text "dati valoarea lui x";
 citeste x;
 scrie x
}
```

Acest exemplu de program afişeaza un numar sau o cifră, dar care mai întâi aşteaptă să fie introdusă valoarea în locatia variabilei x.

4. Un program alcătuit din mai multe instrucţiuni succesive :

```
{text "*****";
 text "**    *";
 text "**    *";
 text "**    *";
 text "**    *";
 text "*****";
 text "**";
```

```
 text "**";
 text "**"
}
```

Scrierea unei litere cu ajutorul steluțelor şi bineînțeles cu ajutorul datelor de tip string.

5. Atribuirea stochează rezultatul unui calcul:

```
{text "dati valoare lui x";
 citeste x;
 text "dati valoare lui y";
 citeste y;
 fie x=x+y;
 scrie x
}
```

În acest program am introdus instrucțiunea de atribuire care stochează rezultatul dorit.

6. Un program poate folosi mai multe variabile, fiecare având locația ei:

```
{text "dati valoare lui x";
 citeste x;
 text "dati valoare lui y";
 citeste y;
 fie x=x+2010;
 scrie x;
 fie y=y+x;
 scrie y
}
```

Alt exemplu în care am folosit diverse instrucțiuni de atribuire.

7. Operații aritmetice şi paranteze în expresii (inclusiv înmulțire):

```
{text "dati valoare lui x";
 citeste x;
 fie y=x+1;
 scrie y;
```

```
 fie z=(x+1)*(y+2);
scrie z
}
```

Un alt exemplu de program în care folosim diverse instrucțiuni de atribuire şi folosind operații aritmetice şi paranteze.

8. A împărți, problema de decizie:

```
{text"dati valoare lui x";
 citeste x;
 text"dati valoare lui y";
 citeste y;
 daca (y==0) atunci text"Impartirea este imposibila"
                altfel { fie z=x/y;
                         scrie z;
                         text "Impartirea a fost efectuata!"
                        }
}
```

Am introdus intrucțiunea de decizie cu ajutorul împărțirii a două numere.

Dacă y este 0 atunci împărțirea nu se poate efectua (se ştie că în matematică numitorul fiind 0, rezultatul este imposibil!!!) .

9. Maximul a două numere:

```
{citeste x;
 citeste y;
 daca (x>=y)
 atunci { daca (x>y) atunci { text "maximul este x"; scrie x}
              altfel  text "numerele sunt egale"    }
 altfel {text "maximul este y"; scrie y }
}
```

În acest program determinăm maximul dintre două numere. În cazul în care sunt egale afişează textul "numerele sunt egale".

10. Maximul a trei numere:

```
{citeste x;
 citeste y;
 citeste z;
 daca (x>=y && x>=z) atunci {text"maximul este x adica"; scrie x} altfel
 daca (y>=x && y>=z) atunci {text"maximul este y adica"; scrie y} altfel
 daca (z>=x && z>=y) atunci {text"maximul este z adica"; scrie z}
}
```

În acest program calculăm maximul a trei numere în care afişează maximul dintre trei numere diferite.

11. Maximul a trei numere folosind *dacă* fără *altfel*:

```
{citeste x;
 citeste y;
 citeste z;
 daca (x>=y && x>=z) atunci {text"maximul este x adica"; scrie x};
 daca (y>=x && y>=z) atunci {text"maximul este y adica"; scrie y};
 daca (z>=x && z>=y) atunci {text"maximul este z adica"; scrie z}
}
```

În acest program ca şi mai sus am calculat de asemenea maximul a trei numere folosind şi instrucţiunea *dacă* .

12. Împărţire, diferit:

```
{ citeste x;
  citeste y;
  daca (y!=0) atunci scrie x/y altfel scrie 0
}
```

Un alt mod de a scrie împărţirea a două numere. Operatorul de comparaţie "*diferit*".

13. Minim sumă, structuri:

```
{text"dati valoare lui x";
 citeste x;
 text"dati valoare lui y";
```

```
 citeste y;
 daca (x<=y) atunci {text "minimul este x";
                     fie x=x+y;
                     scrie x
                    }
             altfel {text "minimul este y";
                     fie y=y+x;
                     scrie y
                    }
}
```

Cerința problemei: Fiind date două variabile *x* şi *y* determinați care conține valoarea minimă şi atribuiți-i ca valoare suma celor două numere

14. Bucla cu test inițial:

```
{fie x=80;
 cat timp(x>50)
    {fie x=x-1;
     scrie x
    };
 text"am numarat descrescator de la 80 la 50!"
}
```

Am numărat descrescător cu ajutorul instrucțiuni *cât timp*, echivalentul lui "*WHILE*" din Pascal sau C. Aceasta este o intrucțiune cu test inițial.

15. Sumă, şir numere terminat cu zero:

```
{fie suma=0;
 text "dati valoare lui x";
 citeste x;
 cat timp(x>0)
     {fie suma=suma+x;
      citeste x
     };
 scrie suma
```

```
}
```

Cerința: adună un set de numere terminând cu 0.

16. Calcul *n!*

```
{ text "Calculul lui n! pentru n= ...";
  text"dati valoare pentru n";
  citeste n;
  fie x=1;
  fie nr=1;
  cat timp(nr<n)
    { fie nr=nr+1;
      fie x=x*nr
    };
 scrie x
}
```

În acest program am arătat cum cu ajutorul buclei cu test inițial îl putem calcula pe n!.

17. Şir aditiv:

```
{ citeste n;
  fie x1=0;
  fie x2=1;
  cat timp(x1<n)
    { fie x1p=x2;
      fie x2p=x2+x1;
      fie x1=x1p;
      fie x2=x2p;
      scrie x1;
      scrie x2;
      text "----"
    }
}
```

18. Cel mai mare divizor comun:

```
{text"dati valoare lui a";
 citeste a;
 text"dati valoare lui b";
 citeste b;
 fie undeimp=a;
 fie unimp=b;
 repeta
   {fie unrest=undeimp%unimp;
    fie undeimp=unimp;
   fie unimp=unrest}
 pina cand (unimp==0);
 scrie undeimp;
text"undeimp reprezinta cmmdc a lui a si b"
}
```

Programul calculează cel mai mare divizor comun cu ajutorul buclei cu test final.

19. Numărătoarea descendentă:

```
{citeste x;
 citeste y;
 repeta{fie x=x-1;
        scrie x}
 pina cand(x==y)
}
```

Am folosit bucla cu test final pentru a realiza numărarea descendentă.

20. Sumă şir repetat *până când*:

```
{fie s=0;
 citeste x;
 repeta {fie s=s+x; citeste x}
 pina cand(x==0);
 scrie s
}
```

Problema sumei şirului de numere terminat cu zero dar rezolvată folosind o buclă cu test final.

21. Factorial cu repetă *până când*:

```
{ text "Calculul lui n! pentru n= ...";
  text"dati valoare pentru n";
  citeste n;
  fie x=1;
  fie nr=1;
  repeta{ fie nr=nr+1;
          fie x=x*nr}
  pina cand(nr==n);
  scrie x
}
```

Calculul lui n! folosind bucla cu test final.

22. Factorial cu execută *atât cât*:

```
{citeste n;
 fie f=1;
 executa {fie f=f*n;
          fie n=n-1
 }
 atat cat(n>1);
 scrie f
}
```

Un alt mod de a calcula n! folosind altă instructiune (execută.....atât cât....).

23. Citirea şi scrierea elementelor unui vector:

```
{text"dati lungimea vectorului ..";
    citeste x;
    pentru (fie i=0; i<x; fie i=i+1)
       {
        fie v[i]=0;
        citeste v[i];
```

```
       text"v[i]=";
       scrie v[i]}
      }
}
```

Acest program calculează mărimea unui vector.

24. Maximul dintr-un vector:

```
{text"dati lungimea vectorului ..";
   citeste x;
   pentru (fie i=0; i<x; fie i=i+1)
      {fie v[i]=0;
       citeste v[i];
       text"v[i]=";
       scrie v[i]
      };
   fie max=v[1];
   pentru (fie i=0; i<x; fie i=i+1)
     {daca (max<v[i]) atunci {max=v[i];
                              scrie max};
                    altfel scrie max
     }
}
```

Calculul maximului dintr-un vector.

25. Acces la elemente unui vector:

```
{ citeste y;
  citeste x;
  fie x[0]=y+x;
  fie x[1]=y*x;
  scrie x[0];
  scrie x[1]
}
```

Program simplu, aplicație pentru Capitolul Vectori.

26. Bucla cu termeni cunoscuți de iterați:

```
{fie s=0;
 citeste n;
 pentru (fie i=0; i<n; fie i=i+1)
    fie s=s+i;
 scrie s;
 text "aceasta este suma numerelor de la 0 la n=.. exclusiv"
}
```

Suma primelor numere. Aici profesorul poate discuta cu elevii săi şi alte metode de rezolvare.

27. Afişare mesaj de sistem şi oprire:

```
{ citeste x;
  citeste y;
  fie z=x/y;
  scrie z;
  text "Stop!"
}
```

Dați *y=0*, împărțirea imposibilă. Mesajul sistemului va fi "*Main: divide by zero*" şi nu mai ajunge să scrie "Stop!".

28. Sumă şi atribuire:

```
{   citeste x1;
    citeste x2;
    fie s=x1+x2;
    scrie s
}
```

29. Sumă şi atribuire:

```
{citeste x1;
 citeste x2;
 citeste x3;
 fie s=x1+x2+x3;
 scrie s
}
```

30. Bucla cu număr cunoscut de iteraţi:

```
{ fie i=0;
   pentru (fie i=1; i<100; fie i=i+1)
   {scrie i}
}
```

31. Triplete Pitagoreice:

```
{ pentru (fie x=1; x<3; fie x=x+1)
  pentru (fie y=x+1; y<3; fie y=y+1)
    { scrie x*x+y*y;
      scrie y*y-x*x;
      scrie 2*x*y
    }
}
```

Calculul primelor triplete de numere Pitagoreice.

32. Împărţirea simulată:

```
{text"introduceti o valoare pentru a";
 citeste a;
 text"introduceti o valoare pentru b";
 citeste b;
 fie x=0;
 fie p=a;
 cat timp(b<=p)
  {fie p=p-b;
   fie x=x+1};
 text "x reprezinta rezultatul impartirii simulate a numarului  a la numarul b";
 scrie x
}
```

Împărţirea simulată prin scaderi repetate.

33. Înjumătăţirea:

```
{citeste x;
 cat timp (x>0)
```

```
   { fie x = x /2;
     scrie x }
}
```

Înjumătățirea până la zero, cu întregi.

34. Maximul dintre două numere:

```
{citeste x;
 citeste y;
 fie max=x;
 daca (max<=y) atunci {fie max=y; scrie max}
              altfel  scrie max
}
```

Altă modalitate de a găsi maximul dintre două numere.

35. Afişarea numărului de divizori:

```
{ fie nrd=0;
    citeste nr;
    pentru (fie d=1; d<nr+1; fie d=d+1)
    { daca (nr%d==0) atunci
        { fie nrd=nrd+1}
    };
  text "Numarul dat are atatia divizori";
  scrie nrd
}
```

Afişează numărul de divizori.

36. Acces la elementele vectorului:

```
{fie x[1]=4;
 fie x[2]=6;
 fie x[3]=8;
 scrie x[1];
 scrie x[2];
```

```
 scrie x[3]
}
```

Concluzii

Puteti folosi limbajul Rodin la predarea tuturor notiunilor care apar in programele de mai sus, beneficiind de software gratuit si de exemple gata facute.

Se pot preda urmatoarele notiuni din materia clasei a IX-a : variabila, constanta, string-ul, intregii, operatii cu intregi, div, mod, atribuire, operatii de I/O, (citire, scriere, scrierea unui text), decizia, alternativa cu doua ramuri, selectia (dupa caz), bucla cu test initial, bucla cu test final (doua variante), bucla „for”, (in romaneste „pentru”) , vectori.