Structură și stil în PROGRAMARE

Bazil Pârv

Continuăm serialul despre stilul în programare cu prezentarea modului în care ar trebui să fie proiectați și implementați algoritmii. Vom descrie modul în care trebuie alese structurile de date, precum și structura generală a unui program.

Toată analiza efectuată în episodul anterior a luat în considerare o problemă *P* generală. Am obținut următoarele:
- patru versiuni finale ale algoritmului de rezolvare a problemei *P*, care nu depind direct de problemă: `1`, `1R1`, `1R2` și `2M`;
- doi subalgoritmi generali, `Continua` și `AfiseazaMeniu`, independenți de problema *P* și apelați în versiunile `1R2` și respectiv `2M`;
- patru subalgoritmi specifici problemei *P*: `CitesteDate`, `CitesteDate2`, `Calculeaza` și `AfiseazaRezultate`, apelați în algoritmii de rezolvare a problemei *P*.

Vom continua să ignorăm enunțul problemei *P*, concentrându-ne atenția pe proiectarea tipurilor de date necesare, stabilirea modului de apelare a subalgoritmilor și pe exprimarea completă a algoritmilor care nu depind de problemă. Pseudocodul folosit în episodul precedent va fi înlocuit de limbajul *Pascal*, fără a pierde nimic din expresivitate.

Înainte de a trece la proiectare și implementare, vom identifica încă un element specific problemei *P*, care nu a fost luat în considerare până acum: ***antetul*** acesteia.

Prin antet înțelegem o scurtă explicație asupra problemei respective, care este afișată de fiecare algoritm înainte de a trece la rezolvarea ei.

Antetului problemei îi va corespunde încă un subalgoritm, a cărui specificare este:

```
SubAlgoritmul Antet este:
   @Afișează un rând ce sintetizează enunțul problemei
SfSubAlgoritm { Antet }
```

Subprogramul care implementează acest algoritm va avea următorul antet:

```
procedure Antet;
```

Proiectarea tipurilor de date

Respectând aceeași manieră ordonată de abordare a rezolvării problemei, ne impunem să definim tipuri de date utilizator pentru fiecare dintre datele evidențiate în analiză. Folosirea acestora ne va permite să exprimăm complet algoritmii și subalgoritmii care sunt în formă finală.

Operațiile meniului

Utilizatorul are de ales între patru operații: `Citire`, `Calcul`, `Afisare` și `Terminare`. Acestea se implementează natural prin tipul de date enumerare:

```
type TipOperatie=
        (Terminare,Citire,Calcul,Afisare);
```

Cu această notație, antetul subprogramului `AfiseazaMeniu` (care implementează subalgoritmul cu același nume) este:

```
procedure AfiseazaMeniu(var op:TipOperatie);
```

Datele de intrare ale problemei

Deoarece nu cunoaștem enunțul problemei, cea mai generală declarare a tipului datelor de intrare este cea de înregistrare:

```
type TipDate = record
     ...   {    aici urmeaza componentele }
           { specifice datelor de intrare }
                      { ale problemei P  }
  end;
```

Să remarcăm că pentru ce ne-am propus în această secțiune nu avem nevoie decât de numele tipului de date `TipDate`.

Antetele subprogramelor `CitesteDate` și `CitesteDate2` (care implementează algoritmii cu același nume) sunt:

```
procedure CitesteDate(var DI:TipDate);
procedure CitesteDate2(var DI:TipDate;
                       var ExistaDate:Boolean);
```

Rezultatele problemei

Declararea tipului rezultatelor este asemănătoare declarării tipului datelor de intrare:

```
type TipRezultate = record
     ...   {     aici urmeaza componentele }
           {        specifice rezultatelor }
           {                  problemei P  }
end;
```

Antetele subprogramelor `Calculeaza` și `AfiseazaRezultate` (care implementează algoritmii cu același nume) sunt:

```
procedure Calculeaza(DI:TipDate;
                     var RC:TipRezultate);
procedure AfiseazaRezultate(DI:TipDate;
                            RC:TipRezultate);
```

Proiectarea și implementarea subalgoritmilor generali

În această secțiune vom prezenta implementarea subalgoritmilor generali `Continuare` și `AfiseazaMeniu`.

Subalgoritmul `Continuare`

Acest subalgoritm este independent de problema *P*. Implementarea în *Pascal* este dată în continuare:

```
function Continuare:Boolean;
      { Pune intrebarea Continuati (D/N)? }
      { si accepta numai una din literele }
      {         'D', 'N', 'd', 'n' pentru }
      {              variabila "raspuns". }
      {    Returneaza true daca "raspuns" }
      {         este 'D' sau 'd' si false }
      {                    in caz contrar }
var raspuns:Char;
            { caracterul care se citeste }
begin
  repeat
     { ne asiguram ca se citeste ce vrem }
    Write('Continuati (D/N)? ');
    Readln(raspuns);
    if not (raspuns in ['D','N','d','n'])
      then Writeln('Nu am inteles!')
  until raspuns in ['D','N','d','n'];
  Continuare:=raspuns in ['D', 'd']
end; { Continuare }
```

Față de descrierea prezentată în cadrul primului episod, implementarea conține verificarea strictă a caracterului citit în variabila `raspuns` (repetând cererea de citire la răspuns incorect) și ignoră diferența dintre literele mari și mici.

Subalgoritmul `AfiseazaMeniu`

Conform descrierii prezentate în primul episod, am păstrat acest algoritm independent de problema *P*. Implementarea sa este prezentată în continuare.

```
procedure AfiseazaMeniu(var op:TipOperatie);
  {  Afiseaza pe ecran meniul programului }
  { si citeste optiunea utilizatorului pe }
  {      care o salveaza in parametrul de }
  {                            iesire "op" }
var o:Char;
begin
  Clrscr;
  repeat
    Writeln('Operatiile posibile sunt: ');
    Writeln('1. Citirea datelor de ',
                                 'intrare');
    Writeln('2. Calcularea rezultatelor');
    Writeln('3. Afisarea rezultatelor');
    Writeln('0. Terminarea executiei ',
                              'programului');
    Write('   Introduceti optiunea ',
                                 '(0-3): ');
       { Citeste o tasta fara a afisa pe }
       {   ecran caracterul corespunzator }
      o:=Readkey
  until o in ['0'..'3'];
    { conversie explicita la TipOperatie }
  op:=TipOperatie(Ord(o)-48);
         { trece la linie noua pe ecran }
  Writeln(Chr(10),Chr(13))
end; { AfiseazaMeniu }
```

Această implementare folosește două subprograme din unit-ul `Crt`: `Clrscr` și `Readkey`. Citirea codului operației dorite se face într-o variabilă caracter, citirea repetându-se dacă nu s-a introdus o valoare corectă. Tasta citită nu este afișată pe ecran, iar operația `op` se obține prin conversia explicită a caracterului citit la tipul `TipOperatie`.

Algoritmi pentru rezolvarea problemei P

În cele ce urmează este prezentată în întregime structura programelor. Textele sursă conțin numai declarațiile de variabile globale și corpurile programelor principale, în cele patru versiuni discutate, locul celorlalte declarații necesare fiind comentat.

Varianta 1

```
Program V1;
       { executie cu un singur set de date }
  { declaratiile tipurilor de date TipDate  }
  { si TipRezultate                         }
  { declaratia subprogramului Antet         }
  { declaratia subprogramului CitesteDate   }
  { declaratia subprogramului Calculeaza    }
```

```
  { declaratia subprogramului                }
  { AfiseazaRezultate                        }
         { declaratia variabilelor globale }
var D:TipDate;          { datele de intrare }
    R:TipRezultate;          { rezultatele }
Begin
  Antet;         { subprogram de prezentare }
  CitesteDate(D);
          { subprogram care citeste datele }
  Calculeaza(D,R);
       { subprogram care face prelucrarile }
  AfiseazaRezultate(D,R);
     { subprogram care afiseaza rezultatele }
  Readln { pentru a putea studia rezultatele }
End. { V1 }
```

Varianta 2 (1R1)

```
Program V2;
      { executie cu seturi repetate de date }
      { datele de intrare dicteaza daca se  }
      { continua sau se opreste executia    }
  { declaratiile tipurilor de date TipDate  }
  { si TipRezultate                         }
  { declaratia subprogramului Antet         }
  { declaratia subprogramului CitesteDate2  }
  { declaratia subprogramului Calculeaza    }
  { declaratia subprogramului               }
  { AfiseazaRezultate                       }
         { declaratia variabilelor globale }
var D:TipDate;          { datele de intrare }
    R:TipRezultate;          { rezultatele }
    SuntDate:Boolean;
       { variabila booleana: true sau false }
Begin
  repeat
    Antet;       { subprogram de prezentare }
    CitesteDate2(D, SuntDate);
          { subprogram care citeste datele }
    if SuntDate then
    begin
      Calculeaza(D,R);
       { subprogram care face prelucrarile }
      AfiseazaRezultate(D,R);
                { subprogram care afiseaza }
                {              rezultatele }
      Readln
       { pentru a putea studia rezultatele }
    end
  until not SuntDate
End. { V2 }
```

Varianta 3 (1R2)

```
Program V3;
  { Executie cu seturi repetate de date.    }
  { Continuarea sau terminarea executiei    }
  { separata de citirea datelor de intrare. }
  { declaratiile tipurilor de date TipDate  }
  { si TipRezultate                         }
  { declaratia subprogramului Antet         }
  { declaratia subprogramului CitesteDate2  }
  { declaratia subprogramului Calculeaza    }
  { declaratia subprogramului               }
  { AfiseazaRezultate                       }
         { declaratia variabilelor globale }
var D:TipDate;          { datele de intrare }
    R:TipRezultate;          { rezultatele }
Begin
  repeat
    Antet;       { subprogram de prezentare }
    CitesteDate2(D);
          { subprogram care citeste datele }
    Calculeaza(D,R);
       { subprogram care face prelucrarile }
    AfiseazaRezultate(D,R);
                    {      subprogram care }
                    { afiseaza rezultatele }
  until not Continuare
End. { V3 }
```

Varianta 4 (2M)

```
Program V4;
  { executie cu seturi repetate de date,    }
  { meniu }
uses Crt;
           { se apeleaza Clrscr si Readkey }
           {              in AfiseazaMeniu }
  { declaratiile tipurilor de date TipDate  }
  { si TipRezultate                         }
  { declaratia tipului de date TipOperatie  }
  { declaratia subprogramului Antet         }
  { declaratia subprogramului CitesteDate2  }
  { declaratia subprogramului Calculeaza    }
  { declaratia subprogramului               }
  { AfiseazaRezultate                       }
  { declaratia subprogramului Mesaj         }
        { declaratia variabilelor globale }
var D: TipDate;         { datele de intrare }
    R: TipRezultate;         { rezultatele }
    operatie:TipOperatie;
                        { operatia ceruta }
    ExistaDate:Boolean;
          { s-au citit datele de intrare? }
    ExistaRezultate:Boolean;
            { s-au calculat rezultatele? }
Begin
  ExistaDate:=false;
  ExistaRezultate:=false;
  repeat
            { afiseaza meniul pe ecran si }
            {            citeste operatia }
    Antet;
    AfiseazaMeniu(operatie);
```

```
    case operatie of
      Citire:
        begin
          CitesteDate(D); { citeste datele }
          ExistaDate:=true;
          ExistaRezultate:=false
        end;
      Calcul:
        if ExistaDate then
          begin
            Calculeaza(D,R);
                      { face prelucrarile }
            Mesaj('Rezultate calculate!');
            ExistaRezultate:=true
          end
        else
          Mesaj('Nu s-au citit '+
                      'datele de intrare!');
      Afisare:
        if ExistaRezultate then
          begin
            AfiseazaRezultate(D, R);
                    { afiseaza rezultatele }
            ReadLn
          end
        else Mesaj('Nu s-au calculat '+
                              'rezultatele!')
    end { case }
  until operatie = Terminare
End. { V4 }
```

Dacă vom compara această implementare cu versiunea 2M, vom constata că s-a prevăzut un mesaj și în situația calculării rezultatelor (utilizatorul este înștiințat că s-a efectuat operația cerută) și că toate mesajele se afișează prin apelul unei proceduri Mesaj al cărei text sursă este prezentat în continuare.

```
  procedure Mesaj(m:string);
        { afiseaza mesajul 'm' si asteapta }
        {              apasarea unei taste }
  var c:Char;
  begin
    Writeln(m);
    Write('Apasati o tasta...');
    c:=ReadKey
  end; { Mesaj }
```

În numărul următor

Ultimul episod al serialului va conține o prezentare a metodelor de implementare a elementelor specifice unei anumite probleme pe care dorim să o rezolvăm.

Vom exemplifica aceste metode pentru o problemă simplă: rezolvarea ecuației de gradul I.

Dl. prof. univ. Bazil Pârv este cadru didactic la Universitatea "Babeș-Bolyai" din Cluj. Poate fi contactat prin e-mail la bparv@cs.ubbcluj.ro.

Noi companii aeriene

Iată ce s-ar întâmpla dacă sistemele de operare ar fi companii aeriene:

DOS AIR

Toți pasagerii vor merge pe pistă, vor împinge avionul până când se desprinde, vor sări în avion, apoi vor sări din nou afară când avionul va atinge din nou pământul, îl vor împinge din nou etc.

WINDOWS AIRLINES

Terminalele sunt foarte curate, personalul are o ținută impecabilă, piloții sunt foarte capabili, iar flota e imensă. Avionul ajunge cu o întârziere de șase luni și după decolare, la înălțimea de 10.000 de metri explodează fără nici un avertisment.

MAC AIRWAYS

Casierii, personajul auxiliar și piloții arată la fel și acționează la fel. Dacă un pasager pune întrebări referitoare la zbor i se spune că nu vrea să știe, nu are nevoie să știe și este rugat să se întoarcă la locul său și să urmărească filmul.

OS/2 SKYWAYS

Terminalul este aproape gol, doar câțiva potențiali pasageri învârtindu-se prin zonă. Personalul le adresează permanent scuze arătându-le avionale de pe pistă. Fiecare potențial pasager află cât de confortabil va fi zborul cu aceste avioane, care sunt motivele pentru care zborul este mai sigur decât în cazul companiei Windows Airlines, dar este rugat să mai aștepte puțin deoarece tehnicienii mai au de lucru la sistemele de zbor.

LINUX EXPRESS

Fiecare pasager va aduce la aeroport o bucată din avion și o cutie cu unelte de lucru. Ei se întâlnesc pe pistă și se ceartă permanent deoarece nu au căzut încă de acord asupra tipului de avion pe care doresc să îl construiască și nici nu știu exact cum să îl asambleze. Până la urmă construiesc mai multe avioane, dar acestea primesc același nume. Unii pasageri chiar ajung la destinație, dar toți pasagerii, fără nici o excepție, sunt convinși că au ajuns unde trebuie.