ANEXE

ANEXA 1
PROGRAME UTILITARE

A1 Aplicații grafice pentru studiul funcțiilor

Atât procesul de cercetare a graficului funcției, cât şi ilustrarea dinamică a metodelor de rezolvare a ecuațiilor algebrice şi transcendente pot fi realizate prin intermediul unor aplicații simple, elaborate în mediul de programare Pascal, cu utilizarea bibliotecii grafice standard **graph.tpu**.

In continuare vor fi prezentate câteva din procedurile şi funcțiile grafice, utile în procesul de realizare a aplicațiilor incluse în biblioteca graph.tpu.

`Detect` – funcția nu are parametri, întoarce un rezultat de tip întreg, care specifică tipul adaptorului video cu care este dotat calculatorul.

`InitGraph` `(var DriverGrafic: integer, var ModGrafic: integer, var DrumSpreDriver : string )`

`DriverGrafic` – indică tipul adaptorului video utilizat. Pentru detectarea tipului se foloseşte de obicei apelul

```
DriverGrafic:= Detect;
```

`ModGrafic` – indică codul erorii apărute la inițializarea regimului grafic. Dacă regimul se inițializează corect, variabila primeşte valoarea 0.

`DrumSpreDriver` – indică calea spre fişierul driverului grafic. Pentru simplitate, copiați fişierul egavga.bgi (driverul grafic) în dosarul de lucru al sistemului TurboPascal. În acest caz parametrul **`DrumSpreDriver`** este un string vid.

Exemplu: procedura regimgrafic din programul `cn020`

`SetColor``(var x: integer)` – procedura fixează culoarea de afişare a elementelor în regim grafic.

`x` – codul culorii. În regim standard pot fi utilizate valori de la 0(negru) la 15(alb)

`OutTextXY` `(x,y: integer; Text: string)` – afişează pe ecran în regim grafic valoarea stringului Text, din punctul cu coordonatele de ecran `x, y`.

`GetMaxX` – Funcția întoarce numărul de pixeli înt-o linie a ecranului în regim grafic.

`GetMaxY` – Funcția întoarce numărul de pixeli înt-o coloană a ecranului în regim grafic.

`Line``(x1,y1,x2,y2: integer)` – Procedura desenează pe ecran în regim grafic o un segment din punctul cu coordonatele de ecran `x1,y1` în punctul cu coordonatele de ecran `x2,y2`.

Notă: Punctul cu coordonatele (0,0) se află în colțul stîng-sus al ecranului grafic, punctul cu coordonatele `(getmaxx, getmaxy)` – în colțul dreapta-jos.

`FillEllipse``(x,y: integer;RazaX, RazaY: integer)` – plasează pe ecran o elipsă cu centrul în punctul cu coordonatele de ecran `x,y` şi razele `RazaX, RazaY`

`Rectangle``(x1,y1,x2,y2: integer)` – trasează un dreptunghi cu vîrfurile diagonale în punctele cu coordonatele de ecran `(x1,y1),(x2,y2)` şi laturile paralele axelor de coordonate.

`Closegraph` – realizează revenirea la regimul text al adaptorului video. (Închide regimul grafic)

Aplicația grafică prezentată în continuare permite vizulizarea graficului funcției de o variabilă, descrise în funcția definită de către utilizator cu numele f. Graficul se construieşte pe intervalul a,b extremitățile căruia se introduc de la tastatură în procesul de rulare al programului**.** Pentru lucrul corect al aplicației trebuie să copiați în dosarul de lucru al sistemului **`TurboPascal`** fişierele **`graph.tpu`** şi **`egavga.bgi`** sau să indicați căile spre aceste fişiere.

```
Program cn020;
uses graph, crt;
const xmax=600;
      ymax=400;
type vreal=array[1..xmax] of real;
var  y:vreal;
     r,h,vmax,vmin,a,b:real;
     i,linie0:integer;
     grDriver: Integer;
     grMode: Integer;
     s:string;
     c:char;

procedure regimgrafic;
begin
   grDriver := Detect;
   InitGraph(grDriver, grMode,'');
   setcolor(15);
end;

function f(x:real):real;
begin
   f:=exp(cos(2*x)*ln(x))+3*sin(x);
   {Aici poate fi descrisa o alta functie}
end;

procedure calcul(a,b:real;var z:vreal);
begin
   h:=abs(b-a)/(xmax-1);
   for i:=1 to xmax do z[i]:=f(a+(i-1)*h);
end;

procedure normare(var z:vreal; var l0:integer; var vmax,vmin:real);
var delta,deplasare:real;
begin
   vmax:=z[1];vmin:=z[1];
   for i:=2 to xmax do
      begin
         if z[i]>vmax then vmax:=z[i];
         if z[i]<vmin then vmin:=z[i];
      end;
   delta:=(vmax - vmin)/(ymax);
   deplasare:=0-vmin;
   l0:=round(ymax-deplasare/delta);
   if vmin>0 then l0:=415;
   if vmax<0 then l0:=10;
   for i:=1 to xmax do
      z[i]:=ymax-(z[i]+deplasare)/delta;
end;

procedure axax(linie0:integer);
begin
   setcolor(11); outtextxy(getmaxx-15,linie0-10,'X');
   line(20,linie0, getmaxx-20,linie0);
   h:=(b-a)/2;
   for i:=0 to 2 do
      begin
         fillellipse(20+i*300,linie0,2,2);
         r:=a+i*h;
         str(r:0:2,s);
         outtextxy(20+i*290,linie0+10,s);
      end;
```

```
end;

procedure axay;
begin
   setcolor(14); outtextxy(10,10,'Y');
   h:=(vmax-vmin);
   for i:=0 to 1 do
      begin
         fillellipse(20,10+i*400,2,2);
         r:=vmax-i*h;
         str(r:0:2,s); outtextxy(20,10+i*400,s);
      end;
   line(20,10,20,getmaxy-65);
end;

procedure modeleazagrafic(y:vreal);
begin
   rectangle(1,1,getmaxx,getmaxy);
   setcolor(11);
   axay;
   axax(linie0);
   for i:=1 to xmax do
      putpixel(i+20,10+round(y[i]),15);
   outtextxy(20,460,'Alt interval (D)a / (N)u');
end;

begin
   repeat
      clrscr;
      write('Introdu extremitatile intervalului: ');
      readln(a,b);
      calcul(a,b,y);
      normare(y,linie0,vmax,vmin);
      regimgrafic;
      modeleazagrafic(y);
      c:=readkey;
      closegraph;
   until upcase(c)='N';
end.
```

A2 Module de generare a determinanților şi sistemelor de ecuații liniare

Crearea şi introducerea elementelor unei matrice *n* x *n* pentru calculul ulterior al determinantului ei cere resurse de timp considerabile, în special dacă se pregătesc teste individuale pentru elevi. Următorul program permite generarea automată a matricelor de dimensiunea cerută şi înscrierea lor în fişierul cu numele indicat de utilizator.

```
Program cn021; {generator de determinanti numerici}
uses crt;
type mat=array[1..20,1..20] of integer;
var n, i,j : integer;
    ss : string;
    a : mat;
    f : text;

function calcul( x:mat;t:integer):real;
var  i,j,k,l:integer;
     s: real;
     minor : mat;
begin
   if t=1 then calcul:=x[1,1]
   else
       begin
```

```
          s:=0;
          for k:=1 to t do
             begin
                for i:=1 to t-1 do
                   for j:=1 to k-1 do
                      minor[i,j]:=x[i+1,j];
                for i:=1 to t-1 do
                   for j:=k to t-1 do
                      minor[i,j]:=x[i+1,j+1];
                if odd(k) then
                    s:=s+x[1,k]*calcul(minor, t-1)
                else
                    s:=s-x[1,k]*calcul(minor, t-1);
             end;
          calcul:=s;
       end;
end;

begin
   clrscr;
   randomize;
   write('Dimensiunea matricei:'); readln(n);
   for i:=1 to n do
      for j:=1 to n do
         a[i,j]:=random(30)-12;
   writeln('Nume:'); readln(ss);
   assign(f,ss); rewrite(f);
   writeln(f,n);
   for i:=1 to n do
      begin
         for j:=1 to n do write(f, a[i,j],' ');
         writeln(f);
      end;
   writeln(f);
   writeln(f, 'Valoare determinant:',calcul(a,n));
   close(f);
end.
```

Elaborarea sistemelor de ecuații şi introducerea manuală a datelor inițiale pentru aceste sisteme este la fel un proces care durează în timp. Următorul modul permite generarea automată a sistemelor de ecuații liniare de dimensiunea indicată de utilizator şi a soluțiilor lor.

```
Program cn022;{generator de sisteme de ecuatii liniare}
uses crt;
type mat=array[1..20,1..20] of integer;
var   n, i, j : integer;
      ss : string;
      b,x : array[1..20] of integer;
      a : mat;
      f : text;

function calcul( x:mat;t:integer):real;
var   i, j, k, l : integer;
      s : real;
      minor : mat;
begin
   if t=1 then calcul:=x[1,1]
   else
       begin
          s:=0;
          for k:=1 to t do
              begin
```

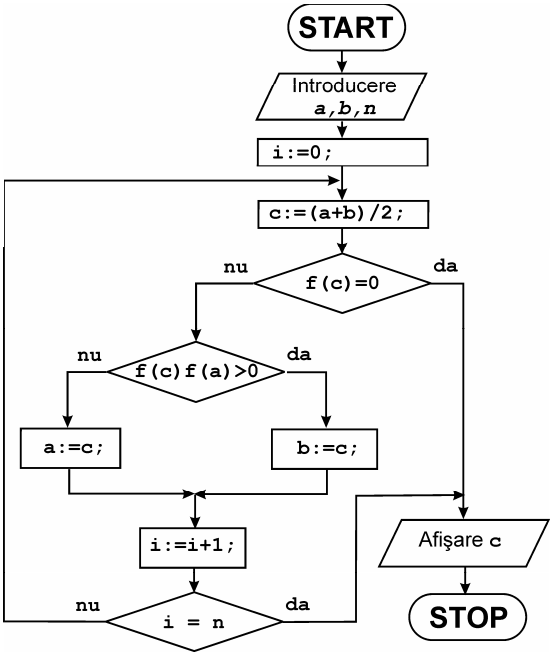
```
                for i:=1 to t-1 do
                   for j:=1 to k-1 do
                      minor[i,j]:=x[i+1,j];
                for i:=1 to t-1 do
                   for j:=k to t-1 do
                      minor[i,j]:=x[i+1,j+1];
                if odd(k) then
                    s:=s+x[1,k]*calcul(minor, t-1)
                else
                    s:=s-x[1,k]*calcul(minor, t-1);
             end;
          calcul:=s;
       end;
end;

begin
   clrscr;
   randomize;
   write('Dimensiunea sistemului:');readln(n);
   for i:=1 to n do x[i]:=random(50)-18;
   for i:=1 to n do
      for j:=1 to n do a[i,j]:=random(50)-21;
   for i:=1 to n do
      begin
         b[i]:=0;
         for j:=1 to n do b[i]:= b[i]+ x[j]*a[i,j];
      end;
   if calcul(a,n)<>0 then
      begin
         writeln('Nume:'); readln(ss);
         assign(f,ss);
         rewrite(f);
         writeln(f,n);
         for i:=1 to n do
            begin
               for j:=1 to n do write(f, a[i,j],' ');
               writeln(f, b[i]);
            end;
         writeln(f);
         writeln(f, 'Solutiile:');
         for i:=1 to n do write(f,x[i]:3,' ');
         close(f);
      end
   else writeln('Determinant nul!');
end.
```

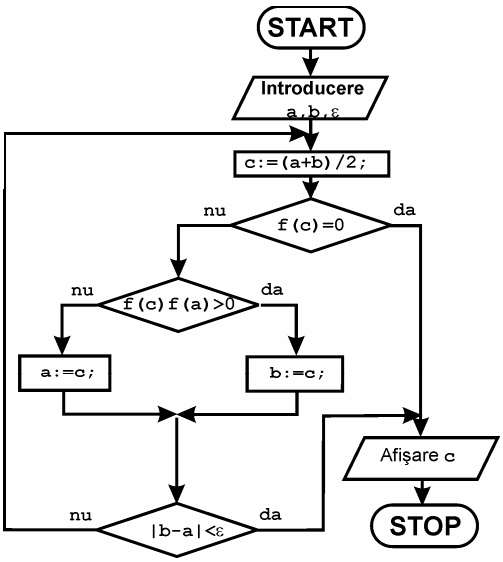
ANEXA 2

SCHEME LOGICE A ALGORITMILOR DE REZOLVARE A ECUAȚIILOR ALGEBRICE ŞI TRANSCENDENTE

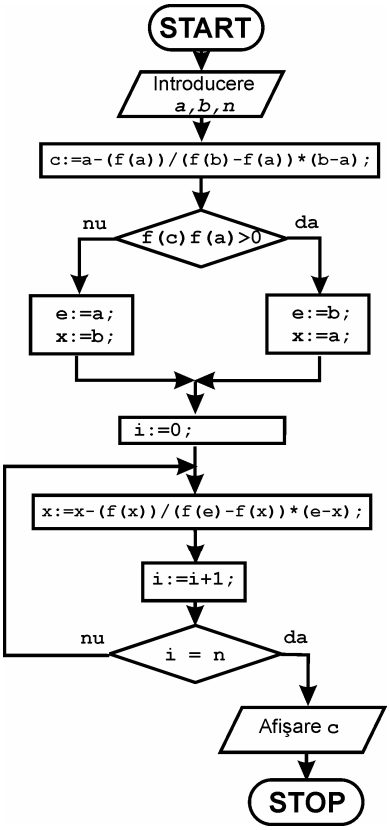
Metoda bisecției – număr fixat de divizări



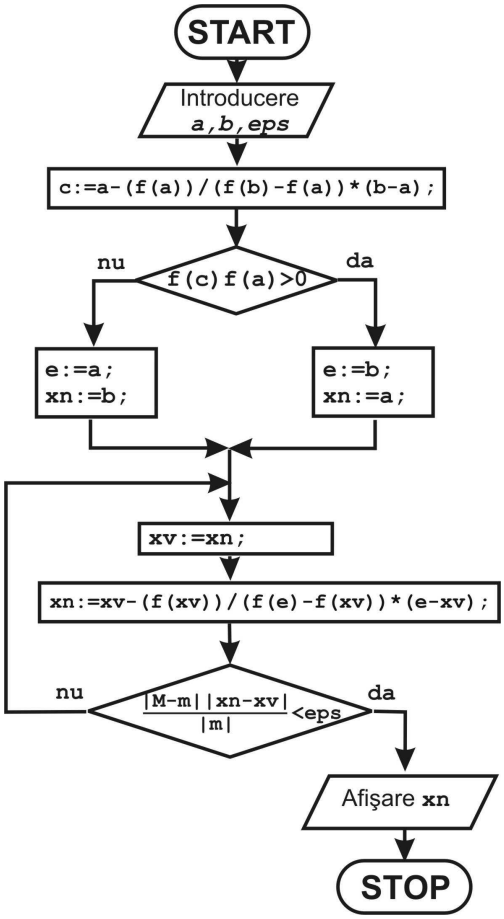
Metoda bisecției – eroare de calcul dată



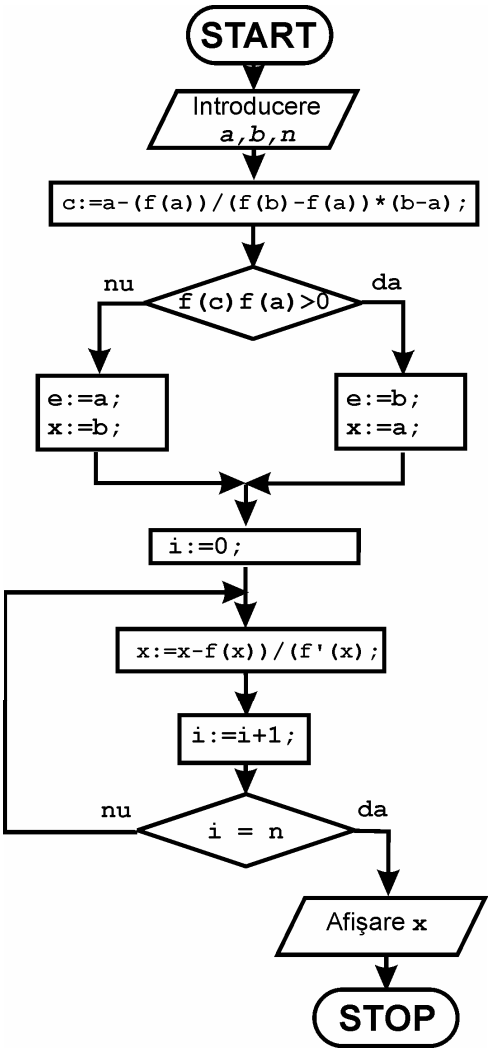
Metoda coardelor – număr fixat de iterații



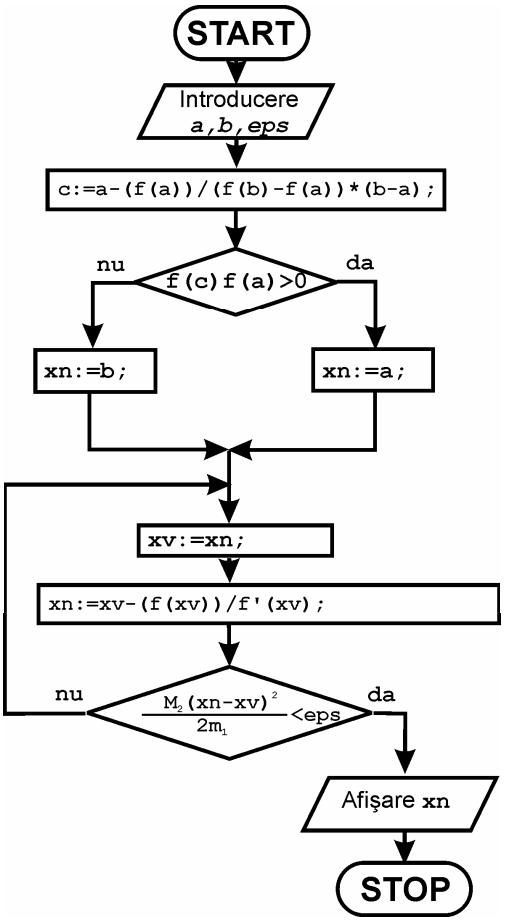
Metoda coardelor – eroare de calcul dată



Metoda tangentelor – număr fixat de divizări



Metoda tangentelor – eroare de calcul dată



Bibliografie

1. Б.П. Демидович, И.А. Марон, *Основы вычислительной математики*, Москва, издательство Физ-Мат литературы, 1963.
2. Б.П. Демидович, *Сборник задач и упражнений по математическому анализу*, Москва, Наука, 1977.
3. L. Gremalschi, Iu. Guzun, *Elemente de modelare*, Chişinău, Lumina, 1995.
4. T. A. Beu, *Calcul numeric în C*, Cluj-Napoca, Editura albastră, 2000.
5. G. D. Mateescu, *Analiza numerică, proiect de manual pentru clasa a XII*. Bucureşti, Editura Petrion, 1995.
6. L. Ammeraal, *Programming Principles in Computer Graphics*, New York, John Wiley & sons, 1992.
7. R. Bradley, *New understanding Computer Science*, London, Nelson Thornes, 2001.