**נספח**

התוכנית הבאה מוצאת מינימום של הפונקציה f(x) = - sin(x) בעזרת "שיטת השליש":

/\* third1.c \*/

#include <stdio.h>

#include <math.h>

long double quarter(long double (\*fp)(long double),

 long double a, long double b, long double eps)

{

 long double x1, fx1, x2, fx2;

 long double factor = 3.0;

do {

 x1 = a + (b-a)/factor;

 fx1 = (\*fp)(x1);

 x2 = b - (b-a)/factor;

 fx2 = (\*fp)(x2);

 if (fx1 > fx2)

 a = x1;

 else

 b = x2;

} while ( (b - a) > eps);

 return ( (a+b)/2);

} /\* quarter \*/

long double f(long double x)

{

 return -sinl(x);

} /\* f \*/

int main()

{

 long double x;

 x = quarter(f, 0.0, 2.0, 0.0001);

 printf("Min solution x = %Lf, min value = %Lf\n", x, f(x));

} /\* main \*/

פלט ריצה:

D:\> third1.exe

Min solution x = 1.570809, min value = -1.000000

D:\>

התוכנית הבאה מוצאת את המינימום של f(x) = -sin(x) בשיטת פיבונצי תוך חישוב n שיתאים לאינטרוול סופי קטן מאפסילון מבוקש:

/\* fibos2.c \*/

#include <stdio.h>

#include <math.h>

void compute\_fibo\_numbers(long double fibo[], long double n)

{

 unsigned int i;

 fibo[0] = fibo[1] = 1.0;

 for (i=2; i <= n; i++)

 fibo[i] = fibo[i-1] + fibo[i-2];

}/\* compute\_fibo\_numbers \*/

void fibo\_search(long double (\*fp)(long double),

 long double a, long double b, unsigned int n,

 long double \*aptr, long double \*bptr)

{

 long double x1, fx1, x2, fx2, fxs;

 long double last, delta;

 long double \*fibo\_n;

 long double fn;

 long double orig\_a, orig\_b, diff;

 int i;

 fibo\_n = (long double\*)malloc((n+1)\*sizeof(long double));

 compute\_fibo\_numbers(fibo\_n, n);

 for(i=0; i <= n; i++)

 printf("fibo[%d] = %Lf\n", i, fibo\_n[i]);

 fn = fibo\_n[n];

 orig\_a = a;

 orig\_b = b;

 diff = b-a;

 x1 = a+diff\*fibo\_n[n-2]/fn;

 x2 = a+diff\*fibo\_n[n-1]/fn;

 fx1 = (\*fp)(x1);

 fx2 = (\*fp)(x2);

 last = x1;

for(i = n-3; i > 0; i--)

{

printf("i = %u\n", i);

printf("a = %Lf, x1 = %Lf, x2 = %Lf, b = %Lf\n", a, x1, x2, b);

 if (fx1 > fx2)

 {

 a = x1;

 x1 = x2;

 fx1 = fx2;

 x2 = b - diff\*fibo\_n[i]/fn;

 fx2 = (\*fp)(x2);

 last = x2;

 } /\* if \*/

 else

 {

 b = x2;

 x2 = x1;

 fx2 = fx1;

 x1 = a + diff\*fibo\_n[i]/fn;

 fx1 = (\*fp)(x1);

 last = x1;

 }/\* else \*/

} /\* for \*/

/\* last comparison \*/

delta = (b-a)/64.0;

if (last == x1)

{

 fxs = (\*fp)(x1+delta);

 if (fxs < fx1)

 {

 \*aptr = x1;

 \*bptr = b;

 } /\* if \*/

 else

 {

 \*aptr = a;

 \*bptr = x1;

 } /\* else \*/

} /\* if \*/

else /\* last == x2 \*/

{

 fxs = (\*fp)(x2-delta);

 if (fxs < fx2)

 {

 \*aptr = a;

 \*bptr = x2;

 } /\* if \*/

 else

 {

 \*aptr = x2;

 \*bptr = b;

 } /\* else \*/

} /\* else \*/

} /\* fibo\_search \*/

long double f(long double x)

{

 return -sinl(x);

} /\* f \*/

int main()

{

 long double x, a, b, eps;

 int n;

 eps = 0.00001;

 n = ((log(2)+log(eps))/log(0.618034))+1;

 printf("n = %d\n", n);

 fibo\_search(f, 0.0, 2.0, n, &a, &b);

 printf("Min solution is between = %Lf to %Lf\n", a, b);

 printf("Min value estimate = %Lf\n", f((a+b)/2.0));

} /\* main \*/

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

C:\> fibos2.exe

n = 23

fibo[0] = 1.000000

fibo[1] = 1.000000

fibo[2] = 2.000000

fibo[3] = 3.000000

fibo[4] = 5.000000

fibo[5] = 8.000000

fibo[6] = 13.000000

fibo[7] = 21.000000

fibo[8] = 34.000000

fibo[9] = 55.000000

fibo[10] = 89.000000

fibo[11] = 144.000000

fibo[12] = 233.000000

fibo[13] = 377.000000

fibo[14] = 610.000000

fibo[15] = 987.000000

fibo[16] = 1597.000000

fibo[17] = 2584.000000

fibo[18] = 4181.000000

fibo[19] = 6765.000000

fibo[20] = 10946.000000

fibo[21] = 17711.000000

fibo[22] = 28657.000000

fibo[23] = 46368.000000

i = 20

a = 0.000000, x1 = 0.763932, x2 = 1.236068, b = 2.000000

i = 19

a = 0.763932, x1 = 1.236068, x2 = 1.527864, b = 2.000000

i = 18

a = 1.236068, x1 = 1.527864, x2 = 1.708204, b = 2.000000

i = 17

a = 1.236068, x1 = 1.416408, x2 = 1.527864, b = 1.708204

i = 16

a = 1.416408, x1 = 1.527864, x2 = 1.596748, b = 1.708204

i = 15

a = 1.527864, x1 = 1.596748, x2 = 1.639320, b = 1.708204

i = 14

a = 1.527864, x1 = 1.570437, x2 = 1.596748, b = 1.639320

i = 13

a = 1.527864, x1 = 1.554175, x2 = 1.570437, b = 1.596748

i = 12

a = 1.554175, x1 = 1.570437, x2 = 1.580487, b = 1.596748

i = 11

a = 1.554175, x1 = 1.564225, x2 = 1.570437, b = 1.580487

i = 10

a = 1.564225, x1 = 1.570437, x2 = 1.574275, b = 1.580487

i = 9

a = 1.564225, x1 = 1.568064, x2 = 1.570437, b = 1.574275

i = 8

a = 1.568064, x1 = 1.570437, x2 = 1.571903, b = 1.574275

i = 7

a = 1.568064, x1 = 1.569531, x2 = 1.570437, b = 1.571903

i = 6

a = 1.569531, x1 = 1.570437, x2 = 1.570997, b = 1.571903

i = 5

a = 1.570437, x1 = 1.570997, x2 = 1.571342, b = 1.571903

i = 4

a = 1.570437, x1 = 1.570782, x2 = 1.570997, b = 1.571342

i = 3

a = 1.570437, x1 = 1.570652, x2 = 1.570782, b = 1.570997

i = 2

a = 1.570652, x1 = 1.570782, x2 = 1.570868, b = 1.570997

i = 1

a = 1.570652, x1 = 1.570738, x2 = 1.570782, b = 1.570868

Min solution is between = 1.570738 to 1.570825

Min value estimate = -1.000000

C:\>

התוכנית הבאה מוצאת מינימום של פונקציה ריבויעית בעזרת שיטת יחס הזהב:

/\* golden.c \*/

#include <stdio.h>

void golden(long double (\*fp)(long double), long double a, long double b,

 long double eps, long double \*aptr, long double \*bptr)

{

 long double x1, fx1, fb, x2, fx2;

 long double phi = 0.61803398874989484820;

 long double phi1 = 1.0 - phi;

 x1 = a + (b-a)\*phi1;

 fx1 = (\*fp)(x1);

 x2 = a + (b-a)\*phi;

 fx2 = (\*fp)(x2);

do {

 if (fx1 > fx2)

 {

 a = x1;

 x1 = x2;

 fx1 = fx2;

 x2 = a + (b-a)\*phi;

 fx2 = (\*fp)(x2);

 }

 else

 {

 b = x2;

 x2 = x1;

 fx2 = fx1;

 x1 = a + (b-a)\*phi1;

 fx1 = (\*fp)(x1);

 } /\* else \*/

} while ( (b - a) > eps);

\*aptr = a;

\*bptr = b;

} /\* golden \*/

long double f(long double x)

{

 return (x-2.5)\*(x-2.5)+ 9.0;

} /\* f \*/

int main()

{

 long double a, b;

 golden(f, -3.0, 7.0, 0.0001, &a, &b);

 printf("Min solution is between = %Lf to %Lf\n", a, b);

 printf("Min value estimate = %Lf\n", f((a+b)/2.0));

} /\* main \*/

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

C:\> golden.exe

Min solution is between = 2.499963 to 2.500060

Min value estimate = 9.000000

C:\>

להלן תוכנית זהה, למעט הדפסות ביניים:

/\* golden1.c \*/

#include <stdio.h>

golden(long double (\*fp)(long double), long double a, long double b,

 long double eps, long double \*aptr, long double \*bptr)

{

 long double x1, fx1, fb, x2, fx2;

 long double phi = 0.61803398874989484820;

 long double phi1 = 1.0 - phi;

 x1 = a + (b-a)\*phi1;

 fx1 = (\*fp)(x1);

 x2 = a + (b-a)\*phi;

 fx2 = (\*fp)(x2);

 printf("a = %Lf, x1 = %Lf, x2 = %Lf, b = %Lf\n", a, x1, x2,b);

do {

 if (fx1 > fx2)

 {

 a = x1;

 x1 = x2;

 fx1 = fx2;

 x2 = a + (b-a)\*phi;

 fx2 = (\*fp)(x2);

 }

 else

 {

 b = x2;

 x2 = x1;

 fx2 = fx1;

 x1 = a + (b-a)\*phi1;

 fx1 = (\*fp)(x1);

 } /\* else \*/

 printf("a = %Lf, x1 = %Lf, x2 = %Lf, b = %Lf\n", a, x1, x2,b);

} while ( (b - a) > eps);

\*aptr = a;

\*bptr = b;

} /\* golden \*/

long double f(long double x)

{

 return (x-2.5)\*(x-2.5)+ 9.0;

} /\* f \*/

int main()

{

 long double a, b;

 golden(f, -3.0, 7.0, 0.0001, &a, &b);

 printf("Min solution is between = %Lf to %Lf\n", a, b);

 printf("Min value estimate = %Lf\n", f((a+b)/2.0));

} /\* main \*/

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

C:\> golden1.exe

a = -3.000000, x1 = 0.819660, x2 = 3.180340, b = 7.000000

a = 0.819660, x1 = 3.180340, x2 = 4.639320, b = 7.000000

a = 0.819660, x1 = 2.278640, x2 = 3.180340, b = 4.639320

a = 0.819660, x1 = 1.721360, x2 = 2.278640, b = 3.180340

a = 1.721360, x1 = 2.278640, x2 = 2.623059, b = 3.180340

a = 2.278640, x1 = 2.623059, x2 = 2.835921, b = 3.180340

a = 2.278640, x1 = 2.491503, x2 = 2.623059, b = 2.835921

a = 2.278640, x1 = 2.410197, x2 = 2.491503, b = 2.623059

a = 2.410197, x1 = 2.491503, x2 = 2.541753, b = 2.623059

a = 2.410197, x1 = 2.460447, x2 = 2.491503, b = 2.541753

a = 2.460447, x1 = 2.491503, x2 = 2.510697, b = 2.541753

a = 2.460447, x1 = 2.479640, x2 = 2.491503, b = 2.510697

a = 2.479640, x1 = 2.491503, x2 = 2.498834, b = 2.510697

a = 2.491503, x1 = 2.498834, x2 = 2.503365, b = 2.510697

a = 2.491503, x1 = 2.496034, x2 = 2.498834, b = 2.503365

a = 2.496034, x1 = 2.498834, x2 = 2.500565, b = 2.503365

a = 2.498834, x1 = 2.500565, x2 = 2.501635, b = 2.503365

a = 2.498834, x1 = 2.499904, x2 = 2.500565, b = 2.501635

a = 2.498834, x1 = 2.499495, x2 = 2.499904, b = 2.500565

a = 2.499495, x1 = 2.499904, x2 = 2.500156, b = 2.500565

a = 2.499495, x1 = 2.499748, x2 = 2.499904, b = 2.500156

a = 2.499748, x1 = 2.499904, x2 = 2.500000, b = 2.500156

a = 2.499904, x1 = 2.500000, x2 = 2.500060, b = 2.500156

a = 2.499904, x1 = 2.499963, x2 = 2.500000, b = 2.500060

a = 2.499963, x1 = 2.500000, x2 = 2.500023, b = 2.500060

Min solution is between = 2.499963 to 2.500060

Min value estimate = 9.000000

C:\>