|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| voenmeh | МИНОБРНАУКИ РОССИИ  федеральное государственное бюджетное  образовательное учреждение высшего образования  **«Балтийский государственный технический университет «ВОЕНМЕХ» им. Д.Ф. Устинова»**  **(БГТУ «ВОЕНМЕХ» им. Д.Ф. Устинова»)** | | | | |
| Факультет | |  | И |  | Информационные и управляющие системы | |
|  | |  | шифр |  | наименование | |
| Кафедра | |  | И9 |  | Систем управления и компьютерных технологий | |
|  | |  | шифр |  | наименование | |
| Дисциплина | |  | Программное обеспечение СРВ | | | |

**Индивидуальное задание №3**

**Определение экстремума целевой функции с использованием генетического алгоритма**

|  |  |  |  |
| --- | --- | --- | --- |
| **ВЫПОЛНИЛ**  студент группы И973 | | | |
| Прокофьев Е.Д. | |
| Фамилия И.О. | | | |
| **ПРЕПОДАВАТЕЛЬ** | | | |
| Куликов Д.Б. | |  | |
| Фамилия И.О. | | | |
|  |  | |  |

САНКТ-ПЕТЕРБУРГ

2021 г.

Выполнение индивидуального задания заключается в реализации простого генетического алгоритма для решения задачи нахождения экстремума целевой функции.

Исходные данные представлены в виде типовых вариантов оптимизируемых целевых функций и перечня параметров используемого алгоритма. Требуется реализовать программную версию заданного алгоритма.

**Типовые варианты обязательного задания**

Найти экстремум целевой функции f(x).

Размер популяции – 50 особей

Длина хромосомы - 20 бит

Область поиска - **[ -3; 3]**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| N  вар  иан  та | Целевая  функция | Искомый  экстре  мум | Способ  кодирования  хромосом | Вероятности | | Генетические  операторы | |
| скрещи  вания  *Pc* | мута  ции  *Pm* | формирование  родительских  пар | редукция |
| 13 | f=-0.9\*x-exp(-0.6\*x).\*sign(cos(9.9\*x+1)) | *min* | двоичный код | 0,85 | 0,2 | Лучшие с лучшими | турнир |

**Текст программы:**

import random as rnd

import matplotlib.pyplot as plt

import numpy as np

from prettytable import PrettyTable

file = open("one.txt","w")

filex = open("onex.txt","w")

filef = open("onef.txt","w")

filefx = open("onefx.txt","w")

filec = open("onec.txt","w")

mytable = PrettyTable()

mytable.field\_names = ["x", "f(x)", "F(x)", "Chromosome"]

L = 20

N = 50

Pc = 0.85

Pm = 0.2

dim = 1#размерность

e = 0.001#точноcть

max\_iteration = 60

min\_iteration = 40

a = [0.0]\*2

b = [0.0]\*2

h = [0.0]\*2

a[0] = -3.0

a[1] = 0.0

b[0] = 3.0

b[1] = 0.0

def strperevod(mas):

stroka = ""

for i in range(len(mas)):

stroka +=str(mas[i])

return stroka

def printShow(k, iter):

print ("Итерация ",iter)

strout = "Итерация " + str(iter) + '\n'

file.write(strout)

filex.write("Итерация "+ str(iter)+ '\n')

filef.write("Итерация "+ str(iter)+ '\n')

filefx.write("Итерация "+ str(iter)+ '\n')

filec.write("Итерация "+ str(iter)+ '\n')

file.write(strout)

for i in range(len(k)):

strout = " x = " + str(k[i].x[0]) + " f(x) = " + str(k[i].f) + " F(x) = " + str(k[i].Fx) + " Хромосома " + strperevod(k[i].chromosome) + '\n'

file.write(strout)

filex.write(str(k[i].x[0])+'\n')

filef.write(str(k[i].f)+'\n')

filefx.write(str(k[i].Fx)+'\n')

filec.write(strperevod(k[i].chromosome)+'\n')

mytable.add\_rows([[str(k[i].x[0]), str(k[i].f), str(k[i].Fx), strperevod(k[i].chromosome)]])

print(mytable)

def Draw(k, iter, flag):

max = -100

min = 100

if (flag==0):

name = "Итерация " + str(iter)

else:

name = "Последняя итерация " + str(iter)

plt.title(name)

x = np.arange(a[0], b[0], 0.001)

y = -0.9\*x-np.exp(0.6\*x)\*np.sign(np.cos(9.9\*x+1))

plt.plot(x, y)

for i in range(len(k)):

plt.plot(k[i].x[0], k[i].f, 'r.')

if (max < k[i].Fx):

max = k[i].Fx

if (min > k[i].Fx):

min = k[i].Fx

printShow(k, iter)

plt.show()

def function(x):#целевая функция

f = -0.9\*x[0]-np.exp(0.6\*x[0])\*np.sign(np.cos(9.9\*x[0]+1))

return f

def functionF(f):#Функция приспособленности

Fx = f - 0.077

return Fx

class individ:

def \_\_init\_\_(self, L, dim):#L-length chromosome, dim-dimension(размерность)

self.L = L

self.chromosome = [0]\*L\*dim

self.dim = dim

self.x= [0.0]\*dim

self.f = 0.0

self.Fx = 0.0

def translateX(self, a1, h1):

count = 0.0

for i in range(self.dim):

for k in range(self.L):

count += (1<<k)\*self.chromosome[k+i\*self.L]

self.x[i] = a1[i] + count\*h1[i]

def func(self):

self.f = function(self.x)

self.Fx = functionF(self.f)

class GAmetod:

def \_\_init\_\_(self,a,b,pc, pm,L,N,dim):

self.a = a

self.b = b

self.pc = pc

self.pm = pm

self.L = L

self.N = N

self.population = list()

self.dim = dim

self.h = [0.0]\*dim

for i in range(dim):

self.h[i] = (b[i]-a[i])/((1<<L)-1)

def gaMetod(self):

dF = 0.0

last\_dF = 0.0

iter = 0

self.generationPopulation()

dF = self.sumF()

while(iter<=max\_iteration):

if (iter == 0 ) or iter == 3:# or iter%10==0):

Draw(self.population, iter, 0)

iter += 1

last\_dF = dF

pair = self.formingPair()

self.crossbreeding(pair)

print(len(self.population))

dF = self.reduction()

if abs((dF-last\_dF)/dF)<e and iter >min\_iteration:

print ("Точность ", abs((dF-last\_dF)/dF))

break

self.iteration = iter

return self.population

def generationPopulation(self):

for j in range(self.N):

per = individ(self.L, self.dim)

for i in range(self.L):

if (rnd.randint(0, 11) % 2 == 0):

per.chromosome[i] = 0

else:

per.chromosome[i] = 1

per.translateX(self.a, self.h)

per.func()

self.population.append(per)

def formingPair(self):

temp = list()

for i in range(self.N):

temp.append(i)

pair = list()

for i in range(int(self.N/2)):

tmp = list()

k = rnd.randint(1,len(temp)) - 1

tmp.append(temp[k])

temp.pop(k)

k = rnd.randint(1, len(temp)) -1

tmp.append(temp[k])

temp.pop(k)

pair.append(tmp)

return pair

def crossbreedingPair(self, one,two):

first = individ(self.L, self.dim)

second = individ(self.L, self.dim)

k = rnd.randint(0, self.L \* self.dim - 1)

for i in range(self.L \* self.dim):

if i < k:

first.chromosome[i] = self.population[one].chromosome[i]

else:

first.chromosome[i] = self.population[two].chromosome[i]

if i > k:

second.chromosome[i] = self.population[one].chromosome[i]

else:

second.chromosome[i] = self.population[two].chromosome[i]

first = self.mutation(first)

second = self.mutation(second)

first.translateX(self.a, self.h)

second.translateX(self.a, self.h)

first.func()

second.func()

self.population.append(first)

self.population.append(second)

def crossbreeding(self,pair):

for i in range(len(pair)):

if (rnd.random() > self.pc):

continue

self.crossbreedingPair(pair[i][0],pair[i][1])

def mutation(self, per):

if (rnd.random() > self.pm):

return per

i = rnd.randint(1, self.L \* self.dim) -1

if (per.chromosome[i] == 1):

per.chromosome[i] = 0

else:

per.chromosome[i] = 1

return per

def sumF(self):

sumFx = 0.0

for i in range(len(self.population)):

sumFx += self.population[i].Fx

return sumFx

def reduction(self):

while(len(self.population)>self.N):

one = rnd.randint(1, len(self.population)) -1

two = rnd.randint(1, len(self.population)) -1

while one == two:

two = rnd.randint(1, len(self.population)) - 1

if (self.population[one].Fx < self.population[two].Fx):

one = two

self.population.pop(one)

return self.sumF()

ga = GAmetod(a,b,Pc,Pm,L,N,dim)

k = ga.gaMetod()

Draw(k,ga.iteration,1)

plt.show()

file.close()

filex.close()

filef.close()

filefx.close()

filec.close()

График целевой функции представлен на рисунке 1.

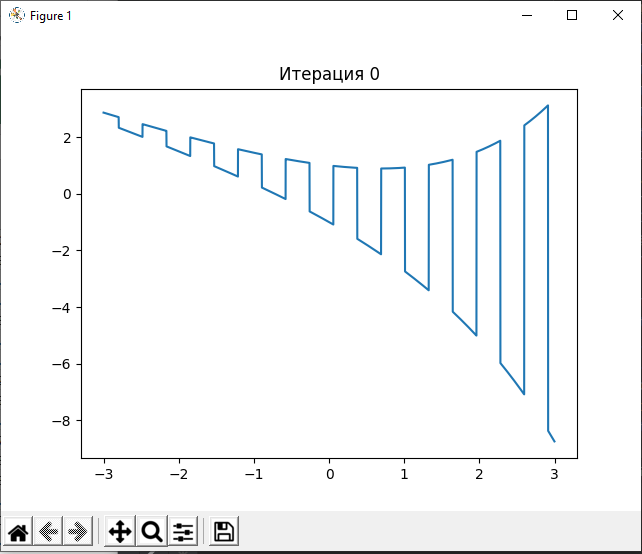


Рисунок 1 – график целевой функции

График функции приспособления представлен на рисунке 2.

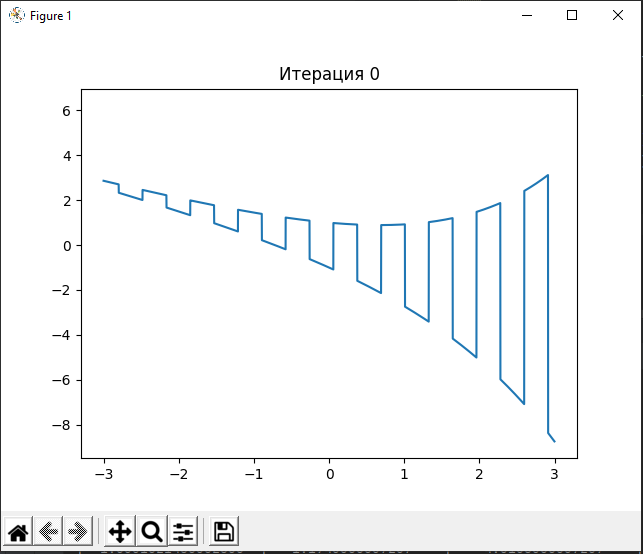


Рисунок 2 – график функции приспособления

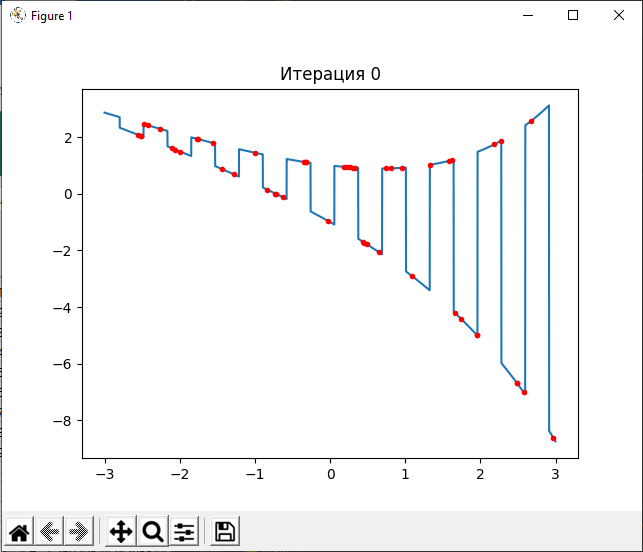
График 0 итерации представлен на рисунке 3.

Рисунок 3 – график 0 итерации

График 3 итерации представлен на рисунке 4.

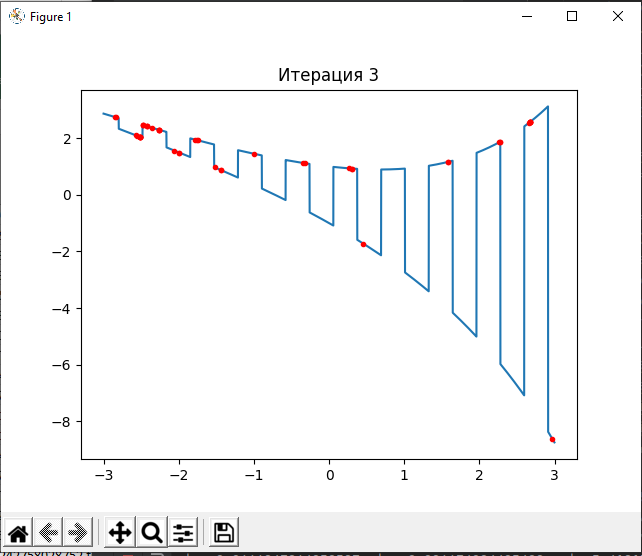


Рисунок 4 – график 3 итерации

График последней итерации представлен на рисунке 5.

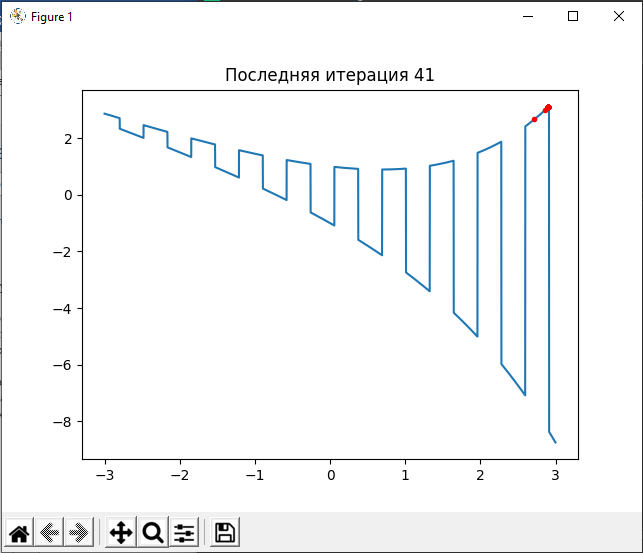


Рисунок 5 – график последней итерации

|  |  |  |  |
| --- | --- | --- | --- |
| ***x*** | ***f(x)*** | ***F(x)*** | ***Хромосома*** |
| **Итерация 0** | **Итерация 0** | **Итерация 0** | **Итерация 0** |
| -2.519622344610543 | 3.103123343762722 | 5.16864154826723 | 00001111111000101000 |
| -2.4221815320792506 | 3.103123343762722 | 5.535258775557514 | 10101110010100011000 |
| 2.1833659967098207 | 3.103123343762722 | 4.862716949571476 | 01000001010010111011 |
| -2.0579214648451467 | 3.103123343762722 | 4.682722675707246 | 00000100110000010100 |
| 0.4924712109291183 | 3.103123343762722 | 1.3345010321979935 | 10001100000010101001 |
| 1.6198812674343754 | 3.103123343762722 | 4.306644191520801 | 01101011100010100011 |
| 1.3299907016665475 | 3.103123343762722 | 4.145590271800879 | 00001111110100011101 |
| 2.67039839782562 | 3.103123343762722 | 5.682276322385723 | 10111111011110001111 |
| 1.9595193476861459 | 3.103123343762722 | -1.882515467231161 | 01001101100111001011 |
| -2.4743036978756887 | 3.103123343762722 | 5.574970316970921 | 00000111011001101000 |
| -0.023903869537229117 | 3.103123343762722 | 2.1572534431594743 | 01110101111101111110 |
| -0.7130562906802087 | 3.103123343762722 | 3.111330890069658 | 00011100100110000110 |
| -1.7562549173878836 | 3.103123343762722 | 5.050756337026372 | 00001000100010101100 |
| 1.5821042843859523 | 3.103123343762722 | 4.28140972801183 | 00110000000111000011 |
| 0.26082540590801795 | 3.103123343762722 | 4.056162334765166 | 01110000010011010001 |
| -0.9995822902510549 | 3.103123343762722 | 4.570073260940195 | 01111001101010101010 |
| -1.9909791860381947 | 3.103123343762722 | 4.610552425633861 | 11001011000011010100 |
| -0.8322008440025748 | 3.103123343762722 | 3.263537826136317 | 01000111111000111010 |
| -1.7754466776339317 | 3.103123343762722 | 5.064037487683617 | 01101111110000101100 |
| 2.268143909591589 | 3.103123343762722 | 4.979818688450443 | 01000110001100000111 |
| 0.7506272798798368 | 3.103123343762722 | 4.014838007097729 | 10110110000000000101 |
| -2.518563765109792 | 3.103123343762722 | 5.167548720369666 | 10010101000100101000 |
| -2.100333309491453 | 2.986711693365585 | 4.728202673118239 | 00110100011001100100 |
| -2.2666847864959587 | 3.103123343762722 | 5.418174294425409 | 00111001001011111000 |
| 1.9581403333094913 | 3.103123343762722 | -1.8785942885276166 | 10000011000111001011 |
| 0.1902839567985124 | 3.103123343762722 | 4.07118752714382 | 01100111100000010001 |
| 1.7405364423145695 | 3.103123343762722 | -1.2864537684265676 | 00101100001001010011 |
| -0.31372004863743674 | 3.103123343762722 | 4.232270510516843 | 10101011100101001110 |
| 2.5749488591660112 | 3.103123343762722 | -3.883781754411121 | 00101011101110110111 |
| 0.2266075388026607 | 3.103123343762722 | 4.0631944643595155 | 01001101010110010001 |
| 0.4560160217437952 | 3.103123343762722 | 1.3963841206843717 | 01110010111011001001 |
| 2.483939632358201 | 3.103123343762722 | -3.552755605766043 | 11001101111110010111 |
| -0.7274186395822904 | 3.103123343762722 | 3.1298507272935754 | 01010110111100000110 |
| -2.548352764466061 | 3.103123343762722 | 5.198267703763227 | 11001010001011001000 |
| 1.665388741863958 | 3.095716398294964 | -1.093548239127494 | 11100111000011100011 |
| 1.0907860668049496 | 3.103123343762722 | 0.2156669219718177 | 00100101000101110101 |
| -0.3230355482440457 | 3.103123343762722 | 4.236037074661498 | 10011110110001001110 |
| 0.6495357985837922 | 3.103123343762722 | 1.0603483000554488 | 01010110110111011001 |
| -0.6198669623059865 | 3.10300747442749 | 2.989970995295675 | 01101011000110100110 |
| -1.4331306773478292 | 3.103123343762722 | 3.9881040703696526 | 01100101101101000010 |
| 0.8143823760818254 | 3.103123343762722 | 4.018636403169747 | 11001111110101000101 |
| 0.43312209427079607 | 3.103123343762722 | 1.4349243954514854 | 10110101111001001001 |
| -0.3454259351977682 | 3.103123343762722 | 4.2451952420018175 | 00001100001010001110 |
| -1.5596910092268077 | 3.103123343762722 | 4.917488101437325 | 00000010111010111100 |
| 2.4876932980473496 | 3.103123343762722 | -3.566142030719364 | 11000010010001010111 |
| 0.9547500178814103 | 3.103123343762722 | 4.035538803664395 | 01100011110100010101 |
| 0.30577211930477066 | 3.103123343762722 | 4.047675932777724 | 10111101000010110001 |
| -1.2745783563407482 | 3.103123343762722 | 3.8031688517790916 | 11000111100110010010 |
| 2.972110721693727 | 3.103123343762722 | -5.502657215927429 | 10101111001101111111 |
| 0.33828481510621566 | 3.103123343762722 | 4.042080472164442 | 11110111011001110001 |
| Итерация 3 | 3.103123343762722 | Итерация 3 | Итерация 3 |
| 0.4560160217437952 | 3.103123343762722 | 1.3963841206843717 | 01110010111011001001 |
| -1.4331306773478292 | 3.103123343762722 | 3.9881040703696526 | 01100101101101000010 |
| -1.4331535655532508 | 3.103123343762722 | 3.9881304816736787 | 01000101101101000010 |
| 0.30576639725341526 | 3.103123343762722 | 4.047676958047638 | 00111101000010110001 |
| 0.30576639725341526 | 3.103123343762722 | 4.047676958047638 | 00111101000010110001 |
| 0.26082540590801795 | 3.103123343762722 | 4.056162334765166 | 01110000010011010001 |
| -0.3454259351977682 | 3.103123343762722 | 4.2451952420018175 | 00001100001010001110 |
| 1.5821042843859523 | 3.103123343762722 | 4.28140972801183 | 00110000000111000011 |
| -0.9995822902510549 | 3.103123343762722 | 4.570073260940195 | 01111001101010101010 |
| -1.9909791860381947 | 3.103123343762722 | 4.610552425633861 | 11001011000011010100 |
| -2.057852800228882 | 2.986711693365585 | 4.6826488923099285 | 00110100110000010100 |
| 2.267674701380445 | 3.103123343762722 | 4.9791432821861825 | 00001000001100000111 |
| 2.268143909591589 | 3.103123343762722 | 4.979818688450443 | 01000110001100000111 |
| -1.7471225234246477 | 3.103123343762722 | 5.0444527046382746 | 00110010111010101100 |
| -1.7572505543237251 | 3.103123343762722 | 5.051444208964815 | 01000110000010101100 |
| -2.518077390744582 | 3.103123343762722 | 5.167046580412427 | 01111111000100101000 |
| -2.518077390744582 | 3.103123343762722 | 5.167046580412427 | 01111111000100101000 |
| -2.518563765109792 | 3.103123343762722 | 5.167548720369666 | 10010101000100101000 |
| -2.518563765109792 | 3.103123343762722 | 5.167548720369666 | 10010101000100101000 |
| -2.548352764466061 | 3.103123343762722 | 5.198267703763227 | 11001010001011001000 |
| -2.2666847864959587 | 3.103123343762722 | 5.418174294425409 | 00111001001011111000 |
| -2.2666847864959587 | 3.103123343762722 | 5.418174294425409 | 00111001001011111000 |
| -2.4225076890065087 | 3.095716398294964 | 5.535506568875578 | 00111100010100011000 |
| -2.4743036978756887 | 3.103123343762722 | 5.574970316970921 | 00000111011001101000 |
| 2.6703240111579998 | 3.103123343762722 | 5.682121716060001 | 00001111011110001111 |
| 2.67039839782562 | 3.103123343762722 | 5.682276322385723 | 10111111011110001111 |
| 2.67039839782562 | 3.10300747442749 | 5.682276322385723 | 10111111011110001111 |
| -2.849275445247121 | 3.103123343762722 | 5.866792338734966 | 10100111011001100000 |
| 2.6702610685930903 | 3.103123343762722 | 5.6819909030461915 | 10100111011110001111 |
| -2.8374193548387097 | 3.103123343762722 | 5.857413622675487 | 10111111011101100000 |
| 2.67039839782562 | 3.103123343762722 | 5.682276322385723 | 10111111011110001111 |
| 2.6703240111579998 | 3.103123343762722 | 5.682121716060001 | 00001111011110001111 |
| -2.4743036978756887 | 3.103123343762722 | 5.574970316970921 | 00000111011001101000 |
| -2.4743036978756887 | 3.103123343762722 | 5.574970316970921 | 00000111011001101000 |
| -2.422547743365997 | 3.103123343762722 | 5.535537000251142 | 10101100010100011000 |
| -2.5678935698447893 | 3.103123343762722 | 5.2183808682733 | 00111111011001001000 |
| -2.4221815320792506 | 3.103123343762722 | 5.535258775557514 | 10101110010100011000 |
| -2.360474930262499 | 3.103123343762722 | 5.488541101899815 | 10101001001011011000 |
| -2.2666847864959587 | 3.103123343762722 | 5.418174294425409 | 00111001001011111000 |
| -2.2666847864959587 | 3.103123343762722 | 5.418174294425409 | 00111001001011111000 |
| -2.519055861526357 | 3.103123343762722 | 5.168056748731271 | 11001010000100101000 |
| -2.518077390744582 | 3.103123343762722 | 5.167046580412427 | 01111111000100101000 |
| -1.7768657463700737 | 3.103123343762722 | 5.065021337573661 | 01111111010000101100 |
| 2.268143909591589 | 3.103123343762722 | 4.979818688450443 | 01000110001100000111 |
| -1.9909334096273514 | 3.103123343762722 | 4.610502909299392 | 11011011000011010100 |
| 1.5822072813103496 | 3.103123343762722 | 4.281476710007199 | 01111000000111000011 |
| 2.654576925827909 | 3.103123343762722 | 5.6496146595857955 | 00001100001010001111 |
| -0.3135140547886417 | 3.103123343762722 | 4.2321875123401105 | 10011111100101001110 |
| -1.5171017809884844 | 3.103123343762722 | 4.084472451572664 | 11001010001011111100 |
| 2.972099277591016 | 2.986711693365585 | -5.502606066026451 | 11001111001101111111 |
| Итерация 54 | 3.103123343762722 | Итерация 54 | Итерация 54 |
| 1.4062484800801087 | 3.103123343762722 | 4.180943907854215 | 11111111111111011101 |
| 2.9062499105929476 | 3.103123343762722 | 6.2246233437627225 | 11111111111111011111 |
| 2.9062499105929476 | 3.103123343762722 | 6.2246233437627225 | 11111111111111011111 |
| 2.9062499105929476 | 3.103123343762722 | 6.2246233437627225 | 11111111111111011111 |
| 2.9062499105929476 | 3.103123343762722 | 6.2246233437627225 | 11111111111111011111 |
| 2.9062499105929476 | 3.103123343762722 | 6.2246233437627225 | 11111111111111011111 |
| 2.9062499105929476 | 3.103123343762722 | 6.2246233437627225 | 11111111111111011111 |
| 2.9062499105929476 | 3.103123343762722 | 6.2246233437627225 | 11111111111111011111 |
| 2.9062499105929476 | 3.103123343762722 | 6.2246233437627225 | 11111111111111011111 |
| 2.9062499105929476 | 3.103123343762722 | 6.2246233437627225 | 11111111111111011111 |
| 2.9062499105929476 | 3.095716398294964 | 6.2246233437627225 | 11111111111111011111 |
| 2.9062499105929476 | 3.103123343762722 | 6.2246233437627225 | 11111111111111011111 |
| 2.9062499105929476 | 3.103123343762722 | 6.2246233437627225 | 11111111111111011111 |
| 2.9062499105929476 | 3.103123343762722 | 6.2246233437627225 | 11111111111111011111 |
| 2.9062499105929476 | 3.10300747442749 | 6.2246233437627225 | 11111111111111011111 |
| 2.9062499105929476 | 3.103123343762722 | 6.2246233437627225 | 11111111111111011111 |
| 2.9062499105929476 | 3.103123343762722 | 6.2246233437627225 | 11111111111111011111 |
| 2.9062499105929476 | 3.103123343762722 | 6.2246233437627225 | 11111111111111011111 |
| 2.9062499105929476 | 3.103123343762722 | 6.2246233437627225 | 11111111111111011111 |
| 2.9062499105929476 | 3.103123343762722 | 6.2246233437627225 | 11111111111111011111 |
| 2.9062499105929476 | 3.103123343762722 | 6.2246233437627225 | 11111111111111011111 |
| 2.9062499105929476 | 3.103123343762722 | 6.2246233437627225 | 11111111111111011111 |
| 2.9062499105929476 | 3.103123343762722 | 6.2246233437627225 | 11111111111111011111 |
| 2.9062499105929476 | 3.103123343762722 | 6.2246233437627225 | 11111111111111011111 |
| 2.9062499105929476 | 3.103123343762722 | 6.2246233437627225 | 11111111111111011111 |
| 2.9062499105929476 | 3.103123343762722 | 6.2246233437627225 | 11111111111111011111 |
| 2.9062499105929476 | 3.103123343762722 | 6.2246233437627225 | 11111111111111011111 |
| 2.9062499105929476 | 3.103123343762722 | 6.2246233437627225 | 11111111111111011111 |
| 2.9062499105929476 | 3.103123343762722 | 6.2246233437627225 | 11111111111111011111 |
| 2.9062499105929476 | 3.103123343762722 | 6.2246233437627225 | 11111111111111011111 |
| 2.9062499105929476 | 3.103123343762722 | 6.2246233437627225 | 11111111111111011111 |
| 2.9062499105929476 | 3.103123343762722 | 6.2246233437627225 | 11111111111111011111 |
| 2.9062499105929476 | 3.103123343762722 | 6.2246233437627225 | 11111111111111011111 |
| 2.8593748658894214 | 3.103123343762722 | 6.108211693365585 | 11111111111110011111 |
| 2.9062499105929476 | 3.103123343762722 | 6.2246233437627225 | 11111111111111011111 |
| 2.9062499105929476 | 3.103123343762722 | 6.2246233437627225 | 11111111111111011111 |
| 2.9062499105929476 | 3.103123343762722 | 6.2246233437627225 | 11111111111111011111 |
| 2.9062499105929476 | 2.986711693365585 | 6.2246233437627225 | 11111111111111011111 |
| 2.9062499105929476 | 3.103123343762722 | 6.2246233437627225 | 11111111111111011111 |
| 2.9062499105929476 | 3.103123343762722 | 6.2246233437627225 | 11111111111111011111 |
| 2.9062499105929476 | 3.103123343762722 | 6.2246233437627225 | 11111111111111011111 |
| 2.9062499105929476 | 3.103123343762722 | 6.2246233437627225 | 11111111111111011111 |
| 2.9062499105929476 | 3.103123343762722 | 6.2246233437627225 | 11111111111111011111 |
| 2.9062499105929476 | 3.103123343762722 | 6.2246233437627225 | 11111111111111011111 |
| 2.9062499105929476 | 3.103123343762722 | 6.2246233437627225 | 11111111111111011111 |
| 2.903320220298977 | 3.103123343762722 | 6.217216398294964 | 11111111101111011111 |
| 2.9062499105929476 | 3.103123343762722 | 6.2246233437627225 | 11111111111111011111 |
| 2.9062499105929476 | 3.103123343762722 | 6.2246233437627225 | 11111111111111011111 |
| 2.9062499105929476 | 3.103123343762722 | 6.2246233437627225 | 11111111111111011111 |
| 2.9062041341821043 | 3.095716398294964 | 6.22450747442749 | 11101111111111011111 |

**Вывод:** в результате выполнения работы был построен простой генетический алгоритм и найден экстремум целевой функции.