MINISTRY OF EDUCATION REPUBLIC OF BELARUS
ESTABLISHMENT OF EDUCATION
"BREST STATE TECHNICHNICAL UNIVERSITY"

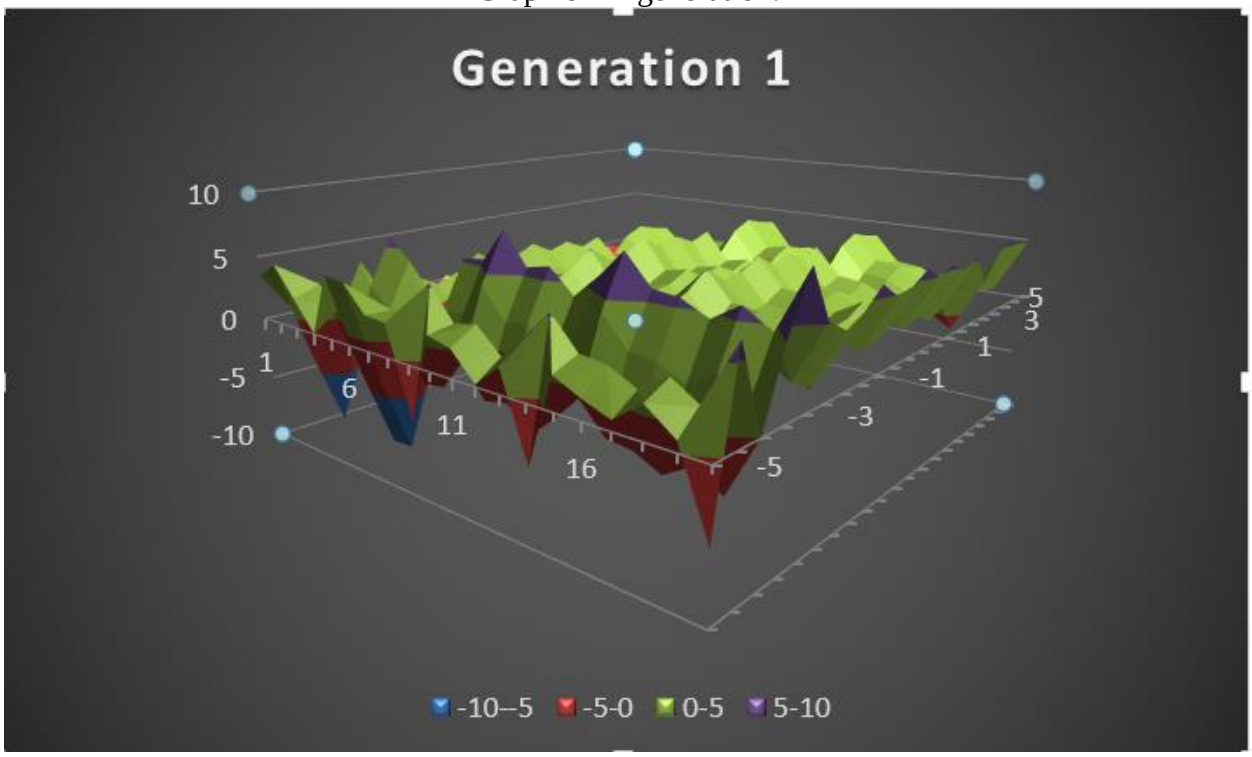
Practice work №2
«Evolutionary Computation»
Subject: «Maximization of 3-d Schefel function»

Made by:
Alexey Cherkasov
Checked by:
Pr. Akira Imada

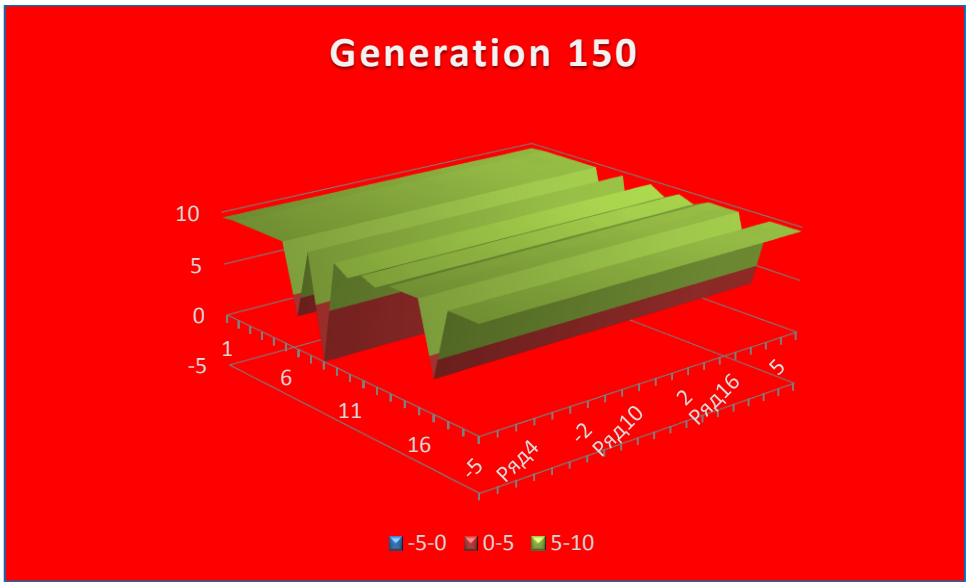
2016

Our function is: y=x1*sin(abs(x1))+x2*sin(abs(x2))

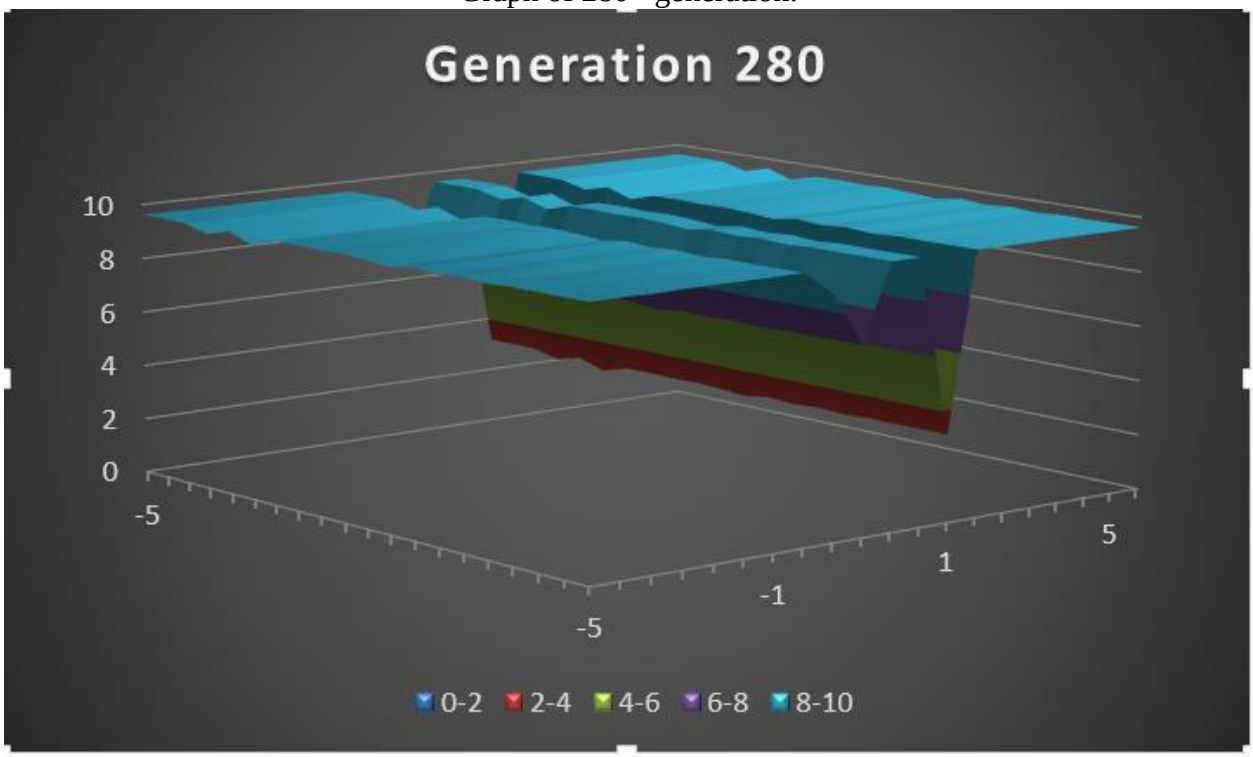
Graph of 1st generation:



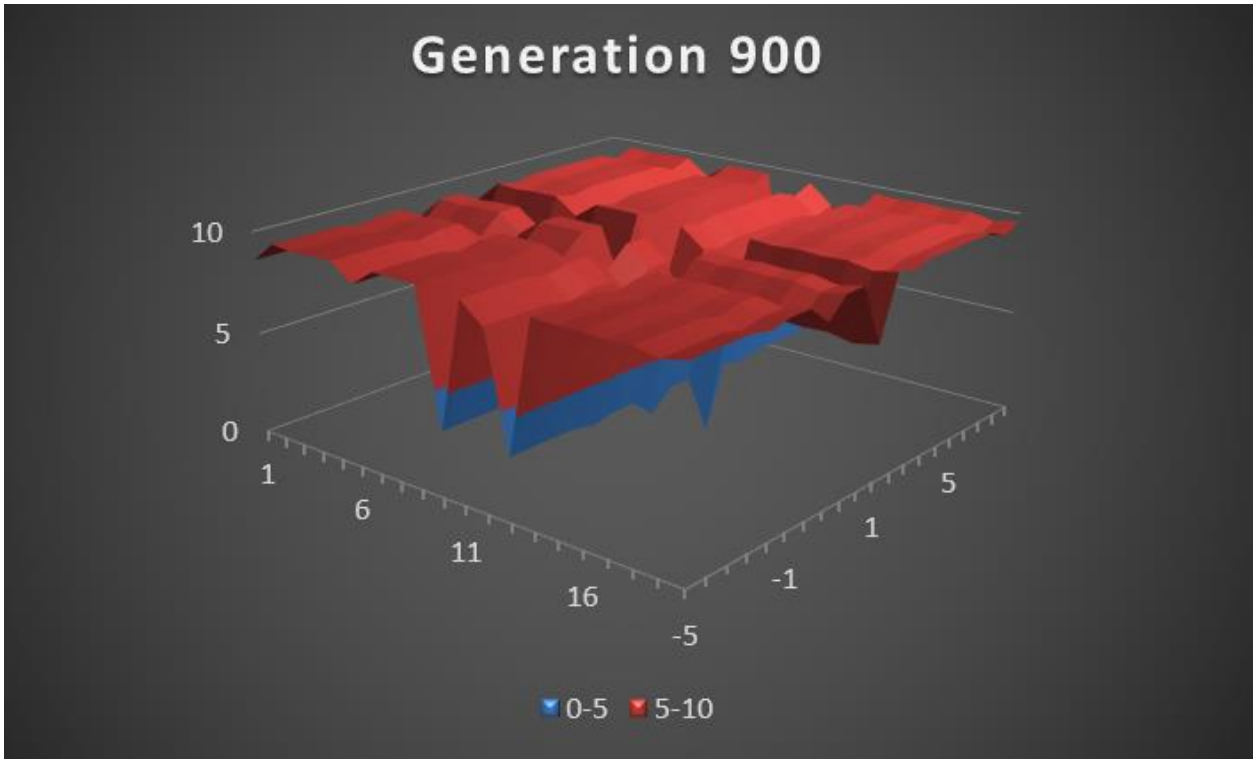
Graph of 1th generation:



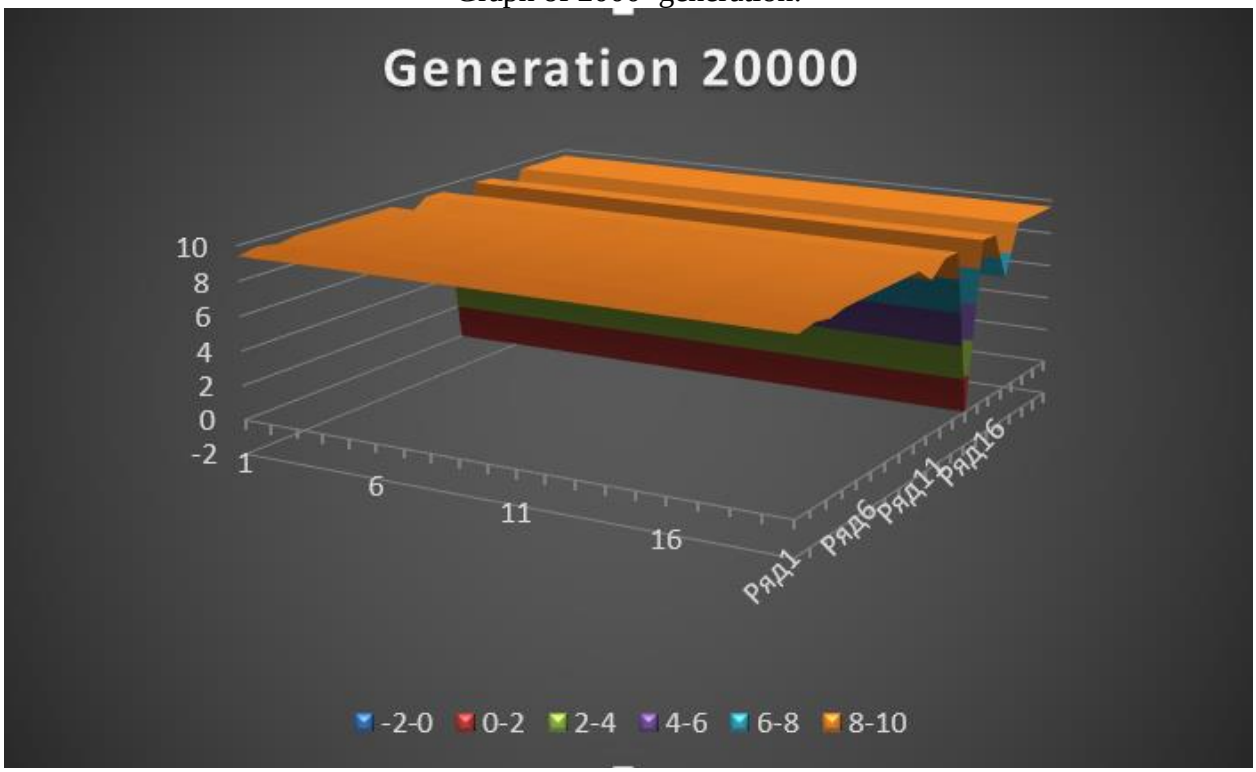
Graph of 280th generation:



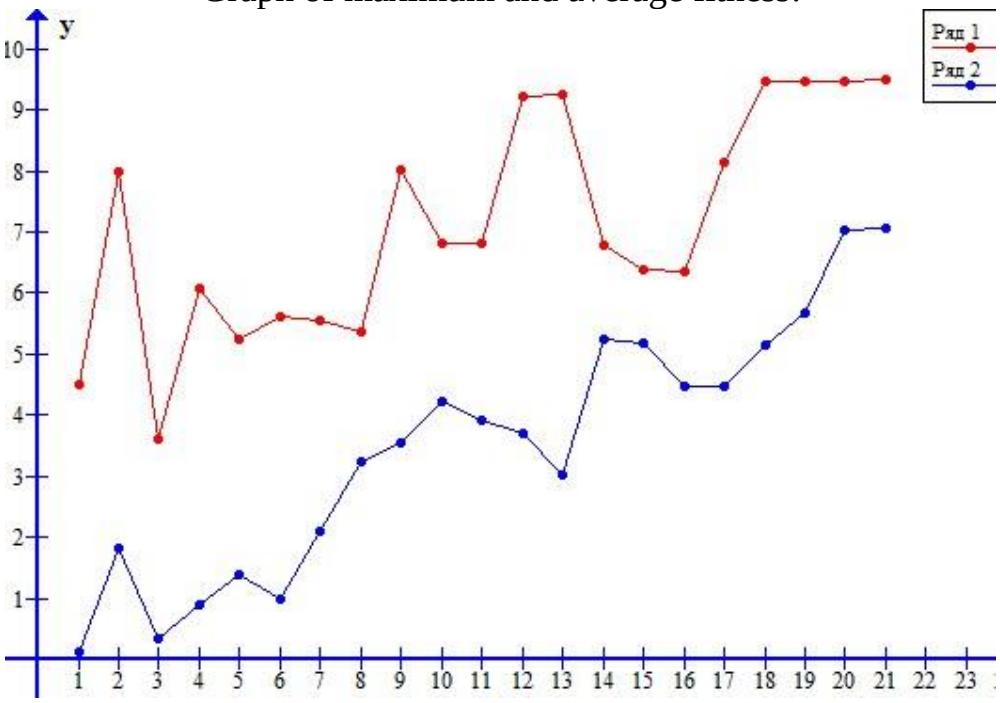
Graph of 900th generation:



Graph of 2000th generation:



Graph of maximum and average fitness:



Program code(C++):

```
#include "stdafx.h"
#include "time.h"
#include <iostream>
#include <fstream>

using namespace std;

int generation[20][11];
double x1[20],x2[20];
double fitness[20];
int gen=0;

ofstream file1;
ofstream file2;
ofstream file3;
ofstream file4;

void bintodec1()
{
	memset(x1,0,20*sizeof(double));
	for(int i=0;i<20;i++)
	{
		for(int j=1;j<11;j++)
		{
			x1[i]+=generation[i][j]*pow(2.0,10-j);
		}
		if(generation[i][0])
			x1[i]=-x1[i];
		x1[i]/=1024;
	}
}
void bintodec2()
{
	memset(x2,0,20*sizeof(double));
	for(int i=0;i<20;i++)
	{
		for(int j=1;j<11;j++)
		{
			x2[i]+=generation[i][j]*pow(2.0,10-j);
		}
		if(generation[i][0])
			x2[i]=-x2[i];
		x2[i]/=1024;
	}
}
void fit()
```

```
{
	for(int i=0;i<20;i++)
		fitness[i]=x1[i]*sin(abs(x1[i]))+x2[i]*sin(abs(x2[i]));
}
void sort()
{
	for(int i=0;i<20;i++)
		for(int j=0;j<20;j++)
			if(i!=j
&&((i<j&&fitness[i]>fitness[j])||(i>j&&fitness[i]<fitness[j])))
			{
				for(int k=0;k<11;k++)
				{
					int gen_t=generation[i][k];
					generation[i][k]=generation[j][k];
					generation[j][k]=gen_t;
				}
				double x1_t = x1[i];
				double x2_t = x2[i];
				x1[i]=x1[j];
				x2[i]=x2[j];
				x1[j]=x1_t;
				x2[j]=x2_t;
				double fit_t = fitness[i];
				fitness[i]=fitness[j];
				fitness[j]=fit_t;
			}
}
void cross()
{
	int new_gener[20][11];
	for(int i=0;i<20;i++)
	{
		int mama=rand()%10;
		int papa=rand()%10;
		int cros_gen=rand()%9+1;
		for(int k=0;k<cros_gen;k++)
			new_gener[i][k]=generation[mama][k];
		for(int j=cros_gen;j<11;j++)
			new_gener[i][j]=generation[papa][j];
	}
	memcpy(generation,new_gener,20*11*sizeof(int));
}
void stats()
{
	file1<<gen<<endl;
	file2<<gen<<endl;
	file3<<gen<<endl;
	file4<<gen<<endl;
	file1<<endl;
	file2<<endl;
	file3<<endl;
	file4<<endl;
	for(int i=0;i<20;i++)
	{
		file1<<x1[i]<<endl;
		file2<<x2[i]<<endl;
		file3<<fitness[i]<<endl;
		for(int j=0;j<11;j++)
			file4<<generation[i][j]<<" ";
		file4<<endl;
	}
	file1<<endl;
	file1<<endl;
	file2<<endl;
	file2<<endl;
	file3<<endl;
	file3<<endl;
	file4<<endl;
	file4<<endl;
	cout<<"min: "<<fitness[0]<<endl;
	//file1<<fitness[0]<<endl;
```

```
	double average=0;
	for(int i=0;i<20;i++)
		average+=fitness[i];
	average/=20;
	cout<<"aver: "<<average<<endl;
	//file2<<average<<endl;
	cout<<"generation: "<<++gen<<endl;
	cout<<"_________________"<<endl;
	/*if(prev_min==fitness[0])
	q++;
	else
	{
	prev_min=fitness[0];
	q=0;
	}
	if(q==15||fitness[0]==1000)
	{
	if(q==15)
	cout<<"there isn't any changes in statistics for 15 generations."<<endl;
	*/
	if(fitness[19]==0)
	{
		file1.close();
		file2.close();
		file3.close();
		file4.close();
		exit(0);
	}
}
void mutation()
{
	//mutations=0;
	for(int i=0;i<20;i++)
		for(int j=0;j<11;j++)
			if(rand()%20==0)
			{
				generation[i][j]=1-generation[i][j];
				//mutations++;
			}
}
int _tmain(int argc, _TCHAR* argv[])
{
	srand(time(NULL));
	file1.open("min1.txt",ios_base::trunc);
	file2.open("min2.txt",ios_base::trunc);
	file3.open("aver.txt",ios_base::trunc);
	file4.open("bin.txt",ios_base::trunc);
	for(int i=0;i<20;i++)
		for(int j=0;j<11;j++)
			generation[i][j]=rand()%2;
	for(int i=0;i<100;i++)
	{
		bintodec1();
		bintodec2();
		fit();
		sort();
		stats();
		cross();
		mutation();
	}
	file1.close();
	file2.close();
	file3.close();
	file4.close();
	return 0;
}
```