

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

Practice work №3
«Evolutionary Computation»
Subject: «Lucky Dog Problem with Fithess Sharing Algorithm»

Made by:
Uladzimir Shukailo
Checked by:
Pr. Akira Imada

1. Create a population of N chromosomes of 1000 genes each of whose values is 1, 2, 3 or 4. Chromosome represents 1000 steps of one dog.

Program code

```
#include <iostream>
#include <Windows.h>
#include <vector>
#include <math.h>
#include <fstream>
#include "time.h"

using namespace std;

void NextG(int mas1[20][1000], int mas2[20][1000]);
void Mut(int mas1[20][1000]);
void Fit(int mas1[20][1000], int F[20]);
void OutCord(int mas[20][1000], int m, int c[20][2]);
void Sh_fit_sort(int mas1[20][1000], int c[20][2], int f[20], int F[20]);
void OutBCAF(vector<int> bc, vector<int> af);
int main()
{
    vector<int> bc;    //
    vector<int> af;
    int ch[20][1000];

    {
        y1 = y1 + f[i];
    }
    af.push_back(y1 / 20);
    cout << bc[0] << ' ';    //display fitness of best chromosome

    for (int end = 0; end == 0;)
    {
        NextG(ch, child);

        Mut(child);
        Fit(child, F);
        Sh_fit_sort(child, c, f, F);
        OutCord(child, g, c);
        bc.push_back(f[0]);    //calculating best and average fitness of generation 1
        y1 = 0;
        for (int i = 0; i < 20; i++)
        {
            y1 = y1 + f[i];
        }
        af.push_back(y1 / 20);
        cout << bc[g] << ' ';    //display fitness of best chromosome 1
        if (bc[g] == 0) end = 1; //stopping 1
        g++;{
            NextG(child, ch);
            Mut(ch);
            Fit(ch, F);
            Sh_fit_sort(ch, c, f, F);
            OutCord(ch, g, c);
            bc.push_back(f[0]);    //calculating best and average fitness of generation 2
            y1 = 0;
            for (int i = 0; i < 20; i++)
            {
                y1 = y1 + f[i];
            }
            af.push_back(y1 / 20);
            cout << bc[g] << ' ';    //display fitness of best chromosome 2
            if (bc[g] == 0) end = 1; //stopping 2
            g++;
        }
    }
    OutBCAF(bc, af);
}
```

```

        std::cout << "(To see more information, check text files: BC.txt, AF.txt, GEN.txt, Xways.txt, Yways.txt)"
<< endl << endl;
        foutX << endl;
        foutY << endl;
        system("pause");
        return 0;
    }

void NextG(int mas1[20][1000], int mas2[20][1000])
{
    srand(time(NULL));
    int x1, x2, e;
    for (int s = 0; s < 20; s++)
    {
        x1 = rand() % 10;
        x2 = rand() % 10;
        for (int i = 0; i < 1000; i++) //uniform crossover
        {
            e = rand() % 2;
            if (e == 1) { mas2[s][i] = mas1[x1][i]; }
            else { mas2[s][i] = mas1[x2][i]; }
        }
    }
}

void Mut(int mas1[20][1000])
{
    srand(time(NULL));
    for (int i = 0; i < 20; i++)
    {
        for (int f = 0; f < 1000; f++)
        {
            int mut = rand() % 20;
            if (mut == 2) {
                mas1[i][f] = rand() % 4 + 1;
            }
        }
    }
}

void Fit(int mas1[20][1000], int F[20])
{
    int X1 = 500, Y1 = 500, x1 = 0, y1 = 0, x2 = 0, y2 = 0, x3 = 0, y3 = 0, x4 = 0, y4 = 0, f1, f2, f3, f4;
    for (int i = 0; i < 20; i++)
    {
        F[i] = 1;
        X1 = 500;
        Y1 = 500;
        for (int f = 0; f < 1000; f++) // dog is moving 1
        {
            switch (mas1[i][f])
            {
                {
                    case 1: { Y1 = Y1 + 1; break; }
                    case 2: { Y1 = Y1 - 1; break; }
                    case 3: { X1 = X1 + 1; break; }
                    case 4: { X1 = X1 - 1; break; }
                    default: cout << "error" << endl;
                }
            }
            if (X1 == 1001) { X1 = 0; }
            if (X1 == -1) { X1 = 1000; }
            if (Y1 == 1001) { Y1 = 0; }
            if (Y1 == -1) { Y1 = 1000; }
            if (X1 == 200 && Y1 == 200) { F[i] = 0; } // if dog find sausage 1
            if (X1 == 200 && Y1 == 800) { F[i] = 0; } // if dog find sausage 2
            if (X1 == 800 && Y1 == 800) { F[i] = 0; } // if dog find sausage 3
            if (X1 == 800 && Y1 == 200) { F[i] = 0; } // if dog find sausage 4
        }
        if (F[i] != 0)
        {
            x1 = 200 - X1;

```

```

        y1 = 200 - Y1;
        x2 = 200 - X1;
        y2 = 800 - Y1;
        x3 = 800 - X1;
        y3 = 800 - Y1;
        x4 = 800 - X1;
        y4 = 200 - Y1;
        f1 = fabs(x1) + fabs(y1);
        f2 = fabs(x2) + fabs(y2);
        f3 = fabs(x3) + fabs(y3);
        f4 = fabs(x4) + fabs(y4);
        if (f1 <= f2 && f1 <= f3 && f1 <= f4) F[i] = f1;
        if (f2 <= f1 && f2 <= f3 && f2 <= f4) F[i] = f2;
        if (f3 <= f2 && f3 <= f1 && f3 <= f4) F[i] = f3;
        if (f4 <= f2 && f4 <= f3 && f4 <= f1) F[i] = f4;
    }
}

void OutCord(int mas[20][1000], int m, int c[20][2])
{
    ofstream foutX("Xways.txt", ios_base::app); //making file of X cordinales
    ofstream foutY("Yways.txt", ios_base::app); //making file of Y cordinales
    int X1 = 500, Y1 = 500;
    foutX << "-" << m + 1 << "-----" << endl;
    foutY << "-" << m + 1 << "-----" << endl;
    for (int i = 0; i < 20; i++)
    {
        foutX << i + 1 << "////" << endl;
        foutY << i + 1 << "////" << endl;
        X1 = 500;
        Y1 = 500;

        for (int f = 0; f < 1000; f++) // dog is moving 1
        {
            switch (mas[i][f])
            {
                case 1: { Y1 = Y1 + 1; break; }
                case 2: { Y1 = Y1 - 1; break; }
                case 3: { X1 = X1 + 1; break; }
                case 4: { X1 = X1 - 1; break; }
                default: cout << "error" << endl;
            }
            foutX << X1 << endl;
            foutY << Y1 << endl;
            c[i][1] = X1;
            c[i][2] = Y1;
        }
        foutX << endl;
        foutY << endl;
    }
}

```