

```

using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace siit_1
{
    class generation
    {
        List<int[]> gens;
        public List<int> fitness { get; }
        public List<float> probability { get; }
        public float averagefitness = 0f;
        int rando = 0;

        public generation()
        {
            gens = new List<int[]>();
            fitness = new List<int>();
            probability = new List<float>();
            for (int j = 0; j < 100; j++)
            {
                int[] gen = new int[1000];
                gens.Add(gen);
                fitness.Add(0);
                probability.Add(0f);
            }
        }

        public generation(List<int[]> new_gens)
        {
            gens = new List<int[]>();
            fitness = new List<int>();
            probability = new List<float>();
            gens = new_gens;
            for (int j = 0; j < 100; j++)
            {
                fitness.Add(0);
                probability.Add(0f);
            }
        }

        public void randomize()
        {
            Random rand = new Random();
            for (int i = 0; i < 100; i++)
            {
                for(int j = 0; j < 1000; j++)
                {
                    gens[i][j] = rand.Next()%2 ;
                }
            }
        }

        public void setFitness()
        {
            for (int i = 0; i < 100; i++)
            {
                for (int j = 0; j < 1000; j++)
                {

```

```

        if (gens[i][j] == 1) fitness[i] += 1;
    }
}
}
public void setProbability()
{
    int mass = 0; ;
    for (int i = 0; i < 100; i++)
    {
        mass += fitness[i];
    }
    averagefitness = mass / 100;
    for (int i = 0; i < 100; i++)
    {
        probability[i] = (float)fitness[i] / (float)mass;
    }
}
public int[] newChild()
{
    Random rand = new Random(DateTime.Now.TimeOfDay.Milliseconds+rando);
    rando++;
    if (rando == 10000000) rando = 0;
    int rand_num = rand.Next(50);
    float sum = 0f;
    int[] chrom_1 = new int[1000], chrom_2 = new int[1000];

    /* for(int i = 0; i < 100; i++)
    {
        sum += probability[i]* 1000000000;
        if (rand_num <= sum)
        {
            chrom_1 = gens[i];
            break;
        }
    }

    */
    chrom_1 = gens[rand_num];
    sum = 0f;
    rand_num = rand.Next(50);
    /*for (int i = 0; i < 100; i++)
    {
        sum += probability[i] * 1000000000;
        if (rand_num <= sum)
        {
            chrom_2 = gens[i];
            break;
        }
    }
    */
    chrom_2 = gens[rand_num];

    //Random lamb = new Random(rand.Next());
    int[] new_chrom = new int[1000];
    for(int i=0;i<1000; i++)
    {
        if (rand.Next()%2 == 1) new_chrom[i] = chrom_1[i];
        else new_chrom[i] = chrom_2[i];
    }
    return new_chrom;
}
}

```

```

    public int bestFitness()
    {
        return fitness.Max();
    }

    public void Sort()
    {
        for(int i = 0; i < 100 - 1; i++)
        {
            bool swapped = false;
            for(int j = 0; j < 100 - i - 1; j++)
            {
                if (fitness[j] < fitness[j + 1])
                {
                    int[] tmp_gen = gens[j];
                    gens[j] = gens[j + 1];
                    gens[j + 1] = tmp_gen;

                    int tmp_fit = fitness[j];
                    fitness[j] = fitness[j + 1];
                    fitness[j + 1] = tmp_fit;
                    swapped = true;
                }
            }
            if (!swapped) break;
        }
    }

    public float getAverageFit()
    {
        return averagefitness;
    }
}

```

```

using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

```

```

namespace siit_1
{
    class Program
    {
        static void Main(string[] args)
        {
            generation old_gens = new generation();
            old_gens.randomize();
            old_gens.setFitness();
            old_gens.setProbability();

            for (int j=0;j<100;)
            {
                Console.WriteLine(old_gens.bestFitness()+" " + old_gens.getAverageFit());
                if (old_gens.bestFitness() == 1000 ) break;
                List<int[]> new_tmp = new List<int[]>();
                old_gens.Sort();
                for (int i = 0; i < 100; i++)

```

```
        {
            new_tmp.Add(old_gens.newChild());
        }
        generation new_gens = new generation(new_tmp);
        new_gens.setFitness();
        if(new_gens.bestFitness()==old_gens.bestFitness())
            old_gens = new_gens;
        old_gens.setFitness();
        old_gens.setProbability();
    }
}
}
```