## (3) 3-D version of Schwefel's function

**Exercise 7** *1. Minimize*

$$y = x_1 \sin(|x_1|) + x_2 \sin(|x_2|)$$

*in the following way!*

*(1) Represent value of $x$ by a 10-bit binary chromosome.*

*(2) Create a population of 20 chromosomes at random, with fitness being $y$.*

*(3) Evolve this population till fitness dosen't change.*

*2. Show*

*(1) the graph of fitness vs generation.*

*(2) all 20 points $(x, y)$ in the 1st, an intermediate, and final generation.*

### Source Code

### Chromosome.cs

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

namespace Graph
{
    public class Chromosome : IComparable<Chromosome>
    {
        public List<int> Genes { get; set; }
        public double x1, x2;
        public double Fitness { get; set; }
        Random rnd;
        const int GENES_NUMBER = 22;

        public Chromosome(Random _rnd)
        {
            rnd = _rnd;
            Genes = new List<int>();

            for (int i = 0; i < GENES_NUMBER; i++)
                Genes.Add(rnd.Next(2));

            CalculateFitness();
        }

        public void CalculateFitness()
        {
            int tempX1, tempX2;

            var x1Genes = Genes.Skip(1).Take(10);
            var x2Genes = Genes.Skip(12);
```

```
        string x1String = String.Empty;
        string x2String = String.Empty;

        foreach (int x in x1Genes)
            x1String += x.ToString();

        foreach (int x in x2Genes)
            x2String += x.ToString();

        tempX1 = Convert.ToInt32(x1String, 2);
        tempX2 = Convert.ToInt32(x2String, 2);

        if (Genes.ElementAt(0) == 0)
            tempX1 = -tempX1;

        if (Genes.ElementAt(11) == 0)
            tempX2 = -tempX2;

        x1 = ((double)tempX1 * 5 / 1023);
        x2 = ((double)tempX2 * 5 / 1023);
       // Console.WriteLine($"x1 = {x1}, x2 = {x2}");
            Fitness = x1 * Math.Sin(Math.Abs(x1)) + x2 * Math.Sin(Math.Abs(x2));
    }

    public int CompareTo(Chromosome compareChromosome)
    {
        if (compareChromosome == null)
            return -1;
        else
            return compareChromosome.Fitness.CompareTo(Fitness);
    }

    public Chromosome CreateChild(Chromosome parent2)
    {
        //uniform crossover
        int chooseParent;
        Chromosome child = new Chromosome(rnd);
        child.Genes.Clear();
        child.Fitness = 0.0;
        int i;
        for (i = 0; i < GENES_NUMBER; i++)
        {
            chooseParent = rnd.Next(2);

            if (chooseParent == 1)
                child.Genes.Add(Genes.ElementAt(i));
            else
                child.Genes.Add(parent2.Genes.ElementAt(i));
        }

        int mutation;
        for (i = 0; i < child.Genes.Count; i++)
        {
            mutation = rnd.Next(22);

            if (mutation == 5)
            {
                if (child.Genes.ElementAt(i) == 1)
                {
                    child.Genes.RemoveAt(i);
                    child.Genes.Insert(i, 0);
                }
                else
                {
                    child.Genes.RemoveAt(i);
```

```
                    child.Genes.Insert(i, 1);
                }
            }
        }

        child.CalculateFitness();

        return child;
    }
  }
}
```

## Program.cs

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

namespace Graph
{
    class Program
    {
        static void Main(string[] args)
        {
            const int CHROMOSOMES_NUMBER = 20;
            Random rnd = new Random();
            int i;
            var chromosomes = new List<Chromosome>();
            var childChromosomes = new List<Chromosome>();
            childChromosomes.Clear();

            double maxFitness, averageFitness;
            int fitnessChange = 0;

            for (i = 0; i < CHROMOSOMES_NUMBER; i++)
                chromosomes.Add(new Chromosome(rnd));

            maxFitness = chromosomes.Max(x => x.Fitness);

            while(fitnessChange != 500)
            {
                chromosomes.Sort();

                for (i = 0; i < CHROMOSOMES_NUMBER; i++)
                {

childChromosomes.Add(chromosomes.ElementAt(rnd.Next(CHROMOSOMES_NUMBER / 2))
                        .CreateChild(chromosomes.ElementAt(rnd.Next(CHROMOSOMES_NUMBER /
2))));
                }

                chromosomes.Clear();
                chromosomes.AddRange(childChromosomes);
                childChromosomes.Clear();

                fitnessChange++;

                if (fitnessChange % 10 == 0)
                {
                    Console.WriteLine("Max Fitness = " + chromosomes.Max(x => x.Fitness)
+ " Average Fitness = " + chromosomes.Average(x => x.Fitness));
```

```
                }
            }
        }
    }
}
```

**Results (Console) 1000 generations**

Max Fitness = 6,63386299132966 Average Fitness = 3,10967426848827

Max Fitness = 9,1155636790317 Average Fitness = 5,46826170532871

Max Fitness = 6,63252461044685 Average Fitness = 4,88506337141749

Max Fitness = 6,63348443280894 Average Fitness = 5,64860141537635

Max Fitness = 6,63303461197515 Average Fitness = 5,53311380585434

Max Fitness = 6,62763644792541 Average Fitness = 4,79486382420858

Max Fitness = 6,63374570647526 Average Fitness = 5,63879221211214

Max Fitness = 6,63402051047885 Average Fitness = 5,24943516115376

Max Fitness = 9,62763118173355 Average Fitness = 3,95116963694149

Max Fitness = 6,63332829461399 Average Fitness = 4,0250574393897

Max Fitness = 6,63363590500649 Average Fitness = 5,02704285461619

Max Fitness = 6,63298471586109 Average Fitness = 4,67592417088847

Max Fitness = 9,37678367194571 Average Fitness = 5,68681891827029

Max Fitness = 9,62884112343752 Average Fitness = 7,09755931166994

Max Fitness = 9,62865079789943 Average Fitness = 8,79113124048653

Max Fitness = 9,62886718905348 Average Fitness = 8,66663550111715

Max Fitness = 9,62884112343752 Average Fitness = 8,70893836392782

Max Fitness = 9,62859238504989 Average Fitness = 8,43222861763103

Max Fitness = 9,62889999544239 Average Fitness = 7,10037071196777

Max Fitness = 9,62889999544239 Average Fitness = 8,2091051547035

Max Fitness = 9,62865079789943 Average Fitness = 8,35507612193925

Max Fitness = 9,62734872505744 Average Fitness = 6,38066714523202

Max Fitness = 9,62604362891081 Average Fitness = 8,46220536624474

Max Fitness = 9,62893280183129 Average Fitness = 8,96602158525466

Max Fitness = 9,62836834122331 Average Fitness = 8,11090089677187

Max Fitness = 9,62781245798769 Average Fitness = 7,54566842637972

Max Fitness = 9,62782150727165 Average Fitness = 9,0347973646002

Max Fitness = 9,6286251914388 Average Fitness = 8,23162187650403

Max Fitness = 9,62782150727165 Average Fitness = 8,43191229817552

Max Fitness = 9,62026332027726 Average Fitness = 8,10200450753791

Max Fitness = 9,6286251914388 Average Fitness = 7,17673611430227

Max Fitness = 9,62836834122331 Average Fitness = 8,17782984944684

Max Fitness = 9,62801959748034 Average Fitness = 7,66209945311883

Max Fitness = 9,62836834122331 Average Fitness = 5,48685513492929

Max Fitness = 9,6269411093096 Average Fitness = 8,50674732051524

Max Fitness = 9,62513322623882 Average Fitness = 7,27939842850734

Max Fitness = 9,62633333679529 Average Fitness = 8,46812211710516

Max Fitness = 9,62649085594449 Average Fitness = 6,94521672719427

Max Fitness = 9,6262160519409 Average Fitness = 8,64488662196413

Max Fitness = 9,62732908519934 Average Fitness = 7,9401524270663

Max Fitness = 9,62513322623882 Average Fitness = 7,71776028921467

Max Fitness = 9,62846001961708 Average Fitness = 7,64752634048208

Max Fitness = 9,62805240386924 Average Fitness = 6,98455608058185

Max Fitness = 9,62652366233339 Average Fitness = 8,15184774304635

Max Fitness = 9,62853351304503 Average Fitness = 8,19113270634631

Max Fitness = 9,62893280183129 Average Fitness = 8,4928555376638

Max Fitness = 9,62751941073743 Average Fitness = 8,74386726436469

Max Fitness = 9,62836834122331 Average Fitness = 6,95431347176556

Max Fitness = 9,62846001961708 Average Fitness = 7,39379793451419

Max Fitness = 9,62836834122331 Average Fitness = 8,0326381990429

Max Fitness = 9,62781245798769 Average Fitness = 6,20999046138552

Max Fitness = 9,62682801876376 Average Fitness = 7,37229384648716

Max Fitness = 9,6286251914388 Average Fitness = 7,54473066270238

Max Fitness = 9,62805240386924 Average Fitness = 7,35565648980054

Max Fitness = 9,62826969407898 Average Fitness = 8,48295408151487

Max Fitness = 9,62751941073743 Average Fitness = 8,07341018032772

Max Fitness = 9,62663901277539 Average Fitness = 7,78855203699169

Max Fitness = 9,62880831704862 Average Fitness = 7,31178754423064

Max Fitness = 9,62889999544239 Average Fitness = 7,87153590536631

Max Fitness = 9,62886718905348 Average Fitness = 8,54154149326041

Max Fitness = 9,62865079789943 Average Fitness = 7,52775085866647

Max Fitness = 9,6283175810463 Average Fitness = 8,57804486907598

Max Fitness = 9,62797762980941 Average Fitness = 7,61924526451927

Max Fitness = 9,6286251914388 Average Fitness = 7,34597053950497

Max Fitness = 9,62751389687915 Average Fitness = 7,81021387458368

Max Fitness = 9,62772982887788 Average Fitness = 7,35021474183627

Max Fitness = 9,62582084226586 Average Fitness = 8,41844299048347

Max Fitness = 9,62826969407898 Average Fitness = 6,6520830313487

Max Fitness = 9,62781245798769 Average Fitness = 8,92751570006055

Max Fitness = 9,62865079789943 Average Fitness = 8,55505946086905

Max Fitness = 9,62826969407898 Average Fitness = 8,78454154722161

Max Fitness = 9,62846001961708 Average Fitness = 6,45792921475113

Max Fitness = 9,62842721322818 Average Fitness = 7,55330120247364

Max Fitness = 9,62836834122331 Average Fitness = 7,87051266068411

Max Fitness = 9,62880831704862 Average Fitness = 8,69637847675411

Max Fitness = 9,62826969407898 Average Fitness = 8,41243692974428

Max Fitness = 9,62643198393963 Average Fitness = 8,48154494642333

Max Fitness = 9,62633333679529 Average Fitness = 7,69207967353298

Max Fitness = 9,62889999544239 Average Fitness = 7,95470155930812

Max Fitness = 9,62893280183129 Average Fitness = 7,2866307724029

Max Fitness = 9,61639788801233 Average Fitness = 7,52225176835881

Max Fitness = 9,62497570708962 Average Fitness = 7,98187646714408

Max Fitness = 9,62649085594449 Average Fitness = 8,4614737941516

Max Fitness = 9,62889999544239 Average Fitness = 7,60196356127244

Max Fitness = 9,62893280183129 Average Fitness = 7,91973040290916

Max Fitness = 9,62870966990429 Average Fitness = 7,2848976971395

Max Fitness = 9,62870966990429 Average Fitness = 8,32946728492901

Max Fitness = 9,62893280183129 Average Fitness = 8,29128181675371

Max Fitness = 9,62889999544239 Average Fitness = 7,29582205526754

Max Fitness = 9,62859238504989 Average Fitness = 8,44480384978462

Max Fitness = 9,62880831704862 Average Fitness = 8,90710971502075

Max Fitness = 9,62124551283938 Average Fitness = 8,48841650988016

Max Fitness = 9,62006991202463 Average Fitness = 7,58275795841115

Max Fitness = 9,62801959748034 Average Fitness = 7,63251565508448

Max Fitness = 9,62786207833114 Average Fitness = 8,54741626808399

Max Fitness = 9,6280949146638 Average Fitness = 7,29343102702464

Max Fitness = 9,62880831704862 Average Fitness = 8,71281234634846

Max Fitness = 9,62714255496595 Average Fitness = 8,35038819322869

Max Fitness = 9,62301811730084 Average Fitness = 8,45797518429507

Max Fitness = 9,62826969407898 Average Fitness = 8,83611846378876