Code of program:

```
public class Main {
    public static Double[][] params = new Double[20][2];
    public static Double[][] generation = new Double[20][10];
    public static Double[][] good = new Double[10][10];
    public static Double[][] childes = new Double[10][10];
    public static Double[] xs = new Double[20];
    public static Integer childCount = 0;
    public static Integer checkCount = 0;
    public static Double prevMax = 0.0;
    public static void main(String[] args) {
        Integer r = 0;
        createGenerate();
        showInfo();
        while(true) {
            Arrays.sort(generation, new Comparator<Double[]>() {
                @Override
                public int compare(Double[] o1, Double[] o2) {
                    Double int1 = fitness(o1);
                    Double int2 = fitness(o2);
                    return (int) (int2 - int1);
                }
            });
            Double max = fitness(generation[0]);
            if(prevMax.equals(max))
              checkCount ++;
            else
                checkCount = 0;
            prevMax = max;
            if(r.equals(5) || r.equals(3) || r.equals(7) || r.equals(0)) {
                System.out.println(returnMaxPrice(generation[0]));
                System.out.println(returnMaxVolume(generation[0]));
                showTable();
            }
            System.out.printf("%.3f %.3f", max,average());
            System.out.println();
            r++;
            if(checkCount > 30) {
                System.out.println(returnMaxPrice(generation[0]));
                System.out.println(returnMaxVolume(generation[0]));
                showTable();
                return;
            }
            setGood();
            Random random = new Random();
            childCount = 0;
```

```
        for (int i = 0; i < 5; i++) {
            createChildes(good[random.nextInt(10)], good[random.nextInt(10)]);
        }
        createNewGeneration();
    }
}

public static void showTable() {
    System.out.println(" It. Pric. Vol.");
    for(int i=0;i<20;i++) {
        System.out.printf("%4d",generation[0][i].intValue());
        System.out.printf("%4d", params[generation[0][i].intValue()][0].intValue());
        System.out.printf("%4d", params[generation[0][i].intValue()][1].intValue());
        System.out.println();
    }
}

public static Double returnY1(Double x) {
    return Math.pow(x - 2,2);
}
public static Double returnY2(Double x) {
    return Math.pow(x - 4,2);
}

public static void createGenerate() {
    Random random = new Random();
    for (int j = 0; j < 10; j++) {
        generation[0][j] = 0.0;
    }
    for (int j = 0; j < 10; j++) {
        generation[19][j] = 1.0;
    }
    for(int i=1;i<19;i++) {
        for (int j = 0; j < 10; j++) {
            generation[i][j] = random.nextInt(2) * 1.0;
        }
    }
}

public static void showInfo() {
    xs = new Double[20];
    for (int t=0;t<20;t++) {
        String str = "";
        for (int i = 0; i < 10; i++) {
            str += String.valueOf(generation[t][i].intValue());
        }
        Double x = Integer.parseInt(str,2) / 170.5;
        xs[t] = x;
        System.out.printf("i=%s x=%.5f y1=%.5f y2=%.5f", str, x, returnY1(x), returnY2(x));
```

```
        System.out.println();
    }
}

public static Double fitness(Double[] obj) {
    Double sumVolume = 0.0;
    Double sumPrice = 0.0;
    for (int i=0; i<obj.length;i++) {
        sumVolume += returnVolume(obj[i]);
    }
    for (int i=0; i<obj.length;i++) {
        sumPrice += returnPrice(obj[i]);
    }
    if(sumVolume < 100)
        return sumPrice * sumVolume;
    else return 0.0;
}

public static Double returnMaxPrice(Double obj[]) {
    Double sumPrice = 0.0;
    for (int i=0; i<obj.length;i++) {
        sumPrice += returnPrice(obj[i]);
    }
    return sumPrice;
}

public static Double returnMaxVolume(Double obj[]) {
    Double sumVolume = 0.0;
    for (int i=0; i<obj.length;i++) {
        sumVolume += returnVolume(obj[i]);
    }
    return sumVolume;
}

public static Double returnPrice(Double item) {
    return params[item.intValue()][0];
}

public static Double returnVolume(Double item) {
    return params[item.intValue()][1];
}

public static void setGood() {
    for(int i =0; i< 10; i++) {
        good[i] = generation[i];
    }
}

public static void createChildes(Double[] parent1, Double[] parent2) {
```

```
        Random random = new Random();
        Integer delimiter = random.nextInt(20);
        Double[] child1 = new Double[20];
        Double[] child2 = new Double[20];
        for(int i = 0;i<20;i++) {
            if(i<=delimiter)
                child1[i] = parent1[i];
            else
                child1[i] = parent2[i];
        }
        for(int i = 0;i<20;i++) {
            if(i<=delimiter)
                child2[i] = parent2[i];
            else
                child2[i] = parent1[i];
        }
        childes[childCount] = child1;
        childCount++;
        childes[childCount] = child2;
        childCount++;
    }

    public static void createNewGeneration() {
        for (int i=0;i<20;i++) {
            if(i<10)
            generation[i] = good[i];
            else
                generation[i] = childes[i-10];
        }
    }

    public static Double average() {
        Double s = 0.0;
        for (int i=0;i<20;i++) {
            s+= fitness(generation[i]);
        }
        Double average = s / 20;
        return average;
    }
}
```