

Министерство образования и науки РФ
Федеральное государственное автономное образовательное учреждение
высшего образования
«Омский государственный технический университет»

Факультет *Информационных технологий и компьютерных систем*
(институт)

Кафедра *Прикладная математика и фундаментальная информатика*

Расчетно–графическая работа

по дисциплине *Алгоритмизация и программирование*

на тему Программная реализация задач

Пояснительная записка

Шифр проекта 020–РГР–02.03.02–№ 7 – ПЗ

Студента *Король Полины Михайловны*
фамилия, имя, отчество полностью

Курс 1 Группа ФИТ-242

Направление (специальность) *02.03.02*
Фундаментальная информатика и информационные технологии
код, наименование

Руководитель *ст. преподаватель*
ученая степень, звание

Федотова И.В.
фамилия, инициалы

Выполнил _____
дата, подпись студента

Работа защищена с количеством баллов	
---	--

дата, подпись руководителя

Омск 2024

Оглавление

Введение	3
Задача 1	4
Постановка задачи 1	4
Ход решения задачи 1	6
Задача 2	10
Постановка задачи 2	10
Ход решения задачи 2	11
Задача 3	13
Постановка задачи 3	13
Ход решения задачи 3	14
Задача 4	16
Постановка задачи 4	16
Ход решения задачи 4	17
Заключение	18
Литература	19

Введение

C# - это объектно-ориентированный язык программирования, разработанный компанией Microsoft, чтобы создавать приложения для Windows. Сегодня C# стал кроссплатформенным, на нем можно писать программы как для Windows, так и для iOS и Linux, особенно веб-приложения и для геймдева. C# относится к семье языков с C-подобным синтаксисом, из них его синтаксис наиболее близок к C++ и Java. Язык имеет статическую типизацию, поддерживает полиморфизм, перегрузку операторов (в том числе операторов явного и неявного приведения типа), делегаты, атрибуты, события, переменные, свойства, обобщённые типы и методы, итераторы, анонимные функции с поддержкой замыканий, LINQ, исключения, комментарии в формате XML.

В данной работе C# был использован для решения четырех задач с применением различных методов данного языка.

Задача 1

Постановка задачи 1

На оптовой базе имеется молоко, выпущенное несколькими фирмами.

Молоко каждой фирмы расфасовано в два вида упаковок, представляющих собой параллелепипеды. Для каждого вида упаковки каждой из фирм известна стоимость, которая включает как стоимость материала тары, так и стоимость собственно молока.

Требуется определить фирму, у которой стоимость одного литра собственно молока минимальна, а также эту стоимость.

Примечание

Считать, что материал тары абсолютно тонкий и все плоскости параллелепипеда состоят из одного слоя материала.

Считать, что у двух упаковок одной фирмы стоимость единицы площади материала одинакова.

Считать, что у двух упаковок одной фирмы стоимость одного литра собственно молока одинакова.

Входной файл

Первая строка содержит целое число N - количество фирм ($1 \leq N \leq 100$).

Следующие N строк содержат шесть целых чисел $X_i^1, Y_i^1, Z_i^1, X_i^2, Y_i^2, Z_i^2$ - размеры двух видов упаковок i -ой фирмы в сантиметрах ($0 < X_i^1, Y_i^1, Z_i^1, X_i^2, Y_i^2, Z_i^2 \leq 100$; $1 \leq i \leq N$), а также два вещественных числа C_i^1 и C_i^2 - стоимости первой и второй упаковок соответственно у i -ой фирмы в рублях ($0 < C_i^1, C_i^2 \leq 1000.0$). В стоимости упаковок включаются как стоимость материала тары, так и стоимость собственно молока.

Выходной файл

Должен содержать одну строку, состоящую из целого и вещественного чисел, разделенных пробелом - номер фирмы, у которой стоимость одного литра собственно молока минимальна, а также эту стоимость в рублях (стоимость выводить с двумя знаками после запятой).

Если имеется несколько фирм с одинаковой минимальной стоимостью собственно молока, то вывести ту из них, номер которой минимален.

Ход решения задачи 1

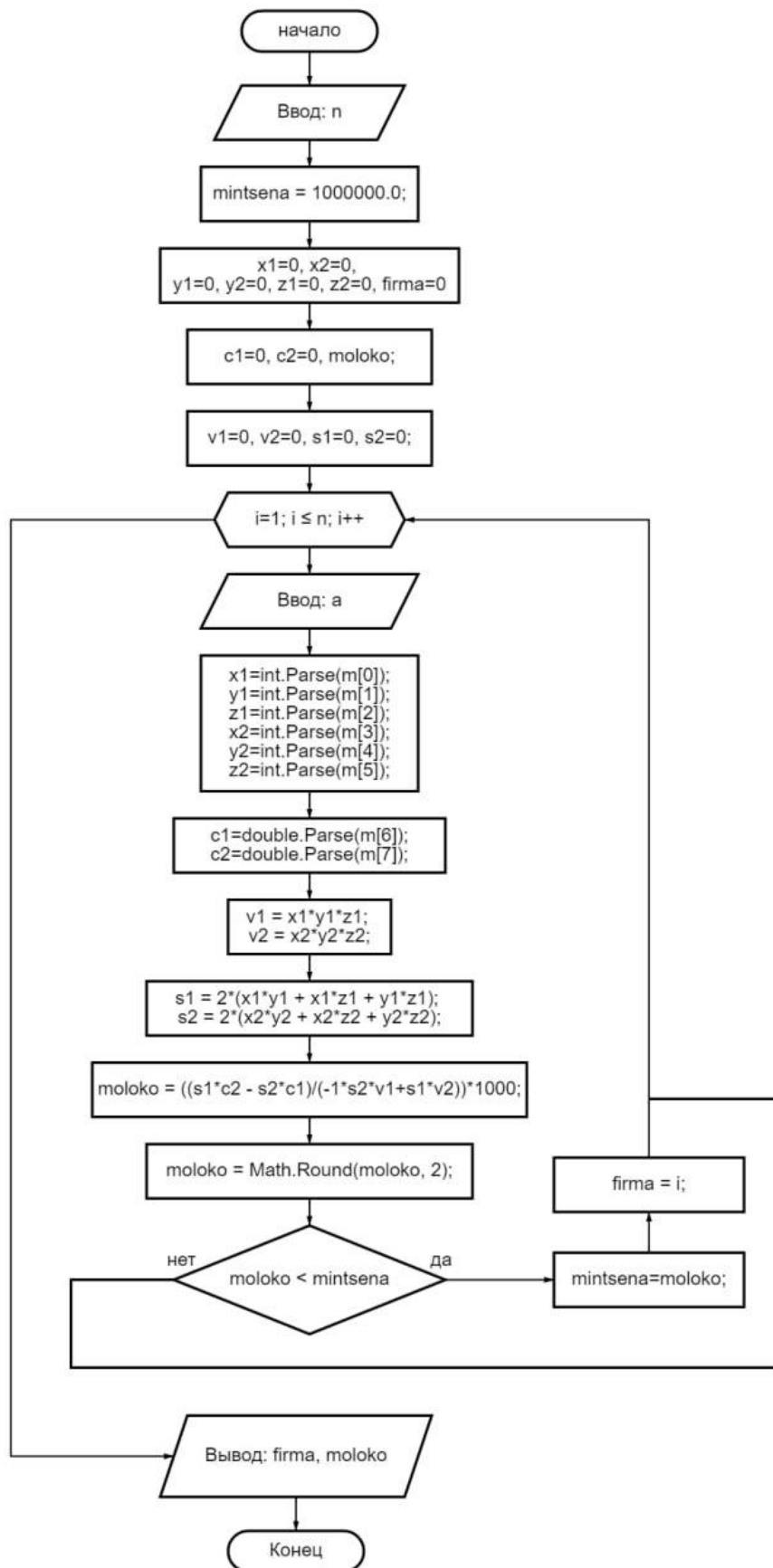


Рисунок 1 – Блок-схема алгоритма задачи 1

Данный алгоритм решает задачу по нахождению фирмы с минимальной стоимостью молока. Программа получает количество фирм и строку с размерами упаковок молока и цен на них для каждой фирмы соответственно. На основе введенных данных об упаковках программа вычисляет стоимость литра молока для каждой из фирм и сравнивает полученные значения, выводя в конце номер фирмы с минимальной ценой молока и саму цену.

```
using System;
```

```
class Program {
```

```
    public static void Main(string[] args) {
```

```
        int n = Convert.ToInt32(Console.ReadLine());
```

```
        double mintsena = 1000000.0;
```

```
        int x1 = 0, y1 = 0, z1 = 0, x2 = 0, y2 = 0, z2 = 0, firma = 0;
```

```
        double c1 = 0, c2 = 0, moloko;
```

```
        int v1 = 0, v2 = 0, s1 = 0, s2 = 0;
```

```
        for(int i = 1; i <= n; i++) {
```

```
            string a = Console.ReadLine();
```

```
            string[] m = a.Split(new char[] { ' ' });
```

```
            x1 = int.Parse(m[0]);
```

```
            y1 = int.Parse(m[1]);
```

```
            z1 = int.Parse(m[2]);
```

```
            x2 = int.Parse(m[3]);
```

```
            y2 = int.Parse(m[4]);
```

```
            z2 = int.Parse(m[5]);
```

```
            c1 = double.Parse(m[6]);
```

```
            c2 = double.Parse(m[7]);
```

```
            v1 = x1*y1*z1;
```

```
            v2 = x2*y2*z2;
```

```
            s1 = 2*(x1*y1 + x1*z1 + y1*z1);
```

```
            s2 = 2*(x2*y2 + x2*z2 + y2*z2);
```

```

moloko = ((s1*c2 - s2*c1)/(-1*s2*v1+s1*v2))*1000;
moloko = Math.Round(moloko, 2);
if(moloko < mintsena) {
    mintsena = moloko;
    firma = i;
}

}

Console.WriteLine($"{ firma} { mintsena}");
}
}

```

```

2
10 10 5 10 10 10 12.23 20.12
5 15 20 7 8 9 43.28 16.99
2 4.17

```

Рисунок 2 – Первый тест задачи 1

```

5
1 1 1 2 2 2 0.59 2.54
1 1 1 2 2 2 0.65 2.92
1 1 1 2 2 2 0.35 1.67
1 1 1 2 2 2 0.39 1.74
1 1 1 2 2 2 0.52 2.11
5 7.5

```

Рисунок 3 – Второй тест задачи 1


```
5
1 1 1 2 2 2 0.28 1.46
1 1 1 2 2 2 0.28 1.46
1 1 1 2 2 2 0.28 1.46
1 1 1 2 2 2 0.28 1.46
1 1 1 2 2 2 0.29 1.47
5 77.5
```

Рисунок 4 – Третий тест задачи 1

Задача 2

Постановка задачи 2

Перед коллективом предприятия “Ни шагу назад” была поставлена задача наращивать каждый день производство продукции на 1.

Требуется определить, какой суммарный объем продукции будет выпущен предприятием за заданный период, если в первый день периода предприятие выпускало P единиц продукции.

Примечания:

- период задается в виде двух календарных дат;
- длительность периода лежит в диапазоне от 1 до 60000;
- високосные годы учитываются по упрощенному правилу: високосным считается год, делящийся нацело на 4;
- день начала периода и день его окончания учитываются при подсчете суммарного объема продукции и длительности периода;
- все даты заданы корректно.

Входной файл содержит:

- в первой строке – дата начала периода в формате ДД.ММ.ГГГГ;
- во второй строке – дата окончания периода в формате ДД.ММ.ГГГГ;
- в третьей строке целое число – начальный выпуск продукции P ($0 \leq P \leq 5000$).

Выходной файл должен содержать суммарный объем продукции.

Ход решения задачи 2

Данный алгоритм решает задачу по вычислению суммарного объема продукции, произведенной на предприятии с даты начала периода по дату окончания периода. Программа получает строку с датой начала периода, строку с датой окончания периода и изначальный объем производимой продукции. В ходе работы программа вычисляет суммарный объем продукции, произведенной за данный период, и выводит его.

```
using System;
```

```
class Program
```

```
{
```

```
    public static void Main(string[] args)
```

```
    {
```

```
        string a = Console.ReadLine();
```

```
        int ayear, byear, amonth, bmonth, aday, bday;
```

```
        string[] ma = a.Split(new char[] { '.' });
```

```
        string b = Console.ReadLine();
```

```
        string[] mb = b.Split(new char[] { '.' });
```

```
        int p = Convert.ToInt32(Console.ReadLine());
```

```
        int s = p;
```

```
        aday = Convert.ToInt32(ma[0]);
```

```
        bday = Convert.ToInt32(mb[0]);
```

```
        amonth = Convert.ToInt32(ma[1]);
```

```
        bmonth = Convert.ToInt32(mb[1]);
```

```
        ayear = Convert.ToInt32(ma[2]);
```

```
        byear = Convert.ToInt32(mb[2]);
```

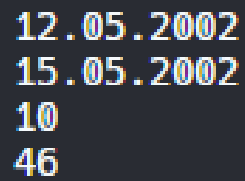
```
        DateTime adate = new DateTime(ayear, amonth, aday);
```

```
        DateTime bdate = new DateTime(byear, bmonth, bday);
```

```
        while(adate != bdate) {
```

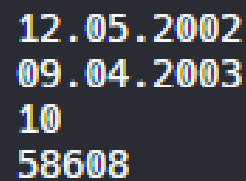
```
            p += 1;
```

```
s += p;  
adate = adate.AddDays(1);  
};  
Console.WriteLine(s);  
}  
}
```



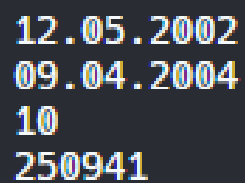
```
12.05.2002  
15.05.2002  
10  
46
```

Рисунок 5 – Первый тест задачи 2



```
12.05.2002  
09.04.2003  
10  
58608
```

Рисунок 6 – Второй тест задачи 2



```
12.05.2002  
09.04.2004  
10  
250941
```

Рисунок 7 – Третий тест задачи 2

Задача 3

Постановка задачи 3

Идет крестьянин и плачется: "Эхма! Жизнь моя горькая! Заела нужда совсем! Вот в кармане только несколько монет, да и те сейчас нужно отдать. И как это у других бывает, что на всякие свои деньги они еще деньги получают? Хотя бы кто помочь мне захотел".

Только успел это сказать, как глядь, а перед ним черт стоит и говорит: "Вот видишь этот мост через реку. Стоит тебе перейти через мост, и у тебя будет вдвое больше денег, чем есть. Перейдешь опять, и снова станет вдвое больше. Но за то, что я у тебя деньги удваиваю, после каждого перехода ты мне должен отдавать по K монет".

"Ой ли," - сказал крестьянин - "ну-ка, попробуем". Перешел мост, и деньги у него удвоились. Отдал он черту K монет, перешел мост еще раз, и опять деньги удвоились. Снова отдал крестьянин черту K монет.

Однако после Z переходов и отдач черту по K монет оказалось, что у крестьянина не осталось ни одной монеты.

Требуется определить, сколько комбинаций условий перехода через мост может быть, если известно, что у крестьянина изначально было не более $MaxN$ монет. Комбинацией условий перехода является тройка чисел N, K, Z , где N - начальное количество монет у крестьянина, K - количество монет, отдаваемых черту после каждого перехода, Z - количество переходов. Естественно, что для этой тройки должно выполняться условие, что после Z циклов у крестьянина не должно остаться монет.

Входной файл содержит целое число $MaxN$ - максимальное количество, которое может быть изначально у крестьянина ($1 \leq MaxN \leq 2000000000$).

Выходной файл должен содержать одно целое число - количество комбинаций условий перехода через мост.

Ход решения задачи 3

Данный алгоритм решает задачу по вычислению количества комбинаций перехода через мост при условии, что у крестьянина было не более заданного числа монет. Программа получает максимальное число монет и в ходе работы подсчитывает все возможные ситуации, когда крестьянин может перейти через мост, после чего выводит их суммарное количество.

```
using System;
```

```
class Program
```

```
{
```

```
    static void Main(string[] args)
```

```
    {
```

```
        long c = 0;
```

```
        long maxn = long.Parse(Console.ReadLine());
```

```
        for(long n = 1; n <= maxn; n++) {
```

```
            for(long k = n + 1; k <= maxn*2; k++) {
```

```
                long n1 = n;
```

```
                while(n1 > 0) {
```

```
                    n1 = 2*n1 - k;
```

```
                }
```

```
                if(n1 == 0) c += 1;
```

```
            }
```

```
        }
```

```
        Console.WriteLine(c);
```

```
    }
```

```
}
```



32
49

Рисунок 8 – Первый тест задачи 3



Рисунок 9 – Второй тест задачи 3



Рисунок 10 – Третий тест задачи 3

Задача 4

Постановка задачи 4

Из N солдат, выстроенных в шеренгу, требуется отобрать троих в разведку. Для того чтобы сделать это, выполняется следующая операция: если солдат в шеренге больше 3, то шеренга разбивается на две, одна из которых состоит из солдат, стоящие на четных позициях, а вторая – стоящих на нечетных позициях. Эта процедура повторяется для всех полученных шеренг до тех пор, пока в каждой из них не останется 3 или менее солдат. Если солдат осталось трое, то данную группу можно послать в разведку.

Требуется определить, сколько групп по 3 человека может быть сформировано из исходной шеренги.

Входной файл содержит число N – количество солдат в исходной шеренге. ($0 < N \leq 10000000$).

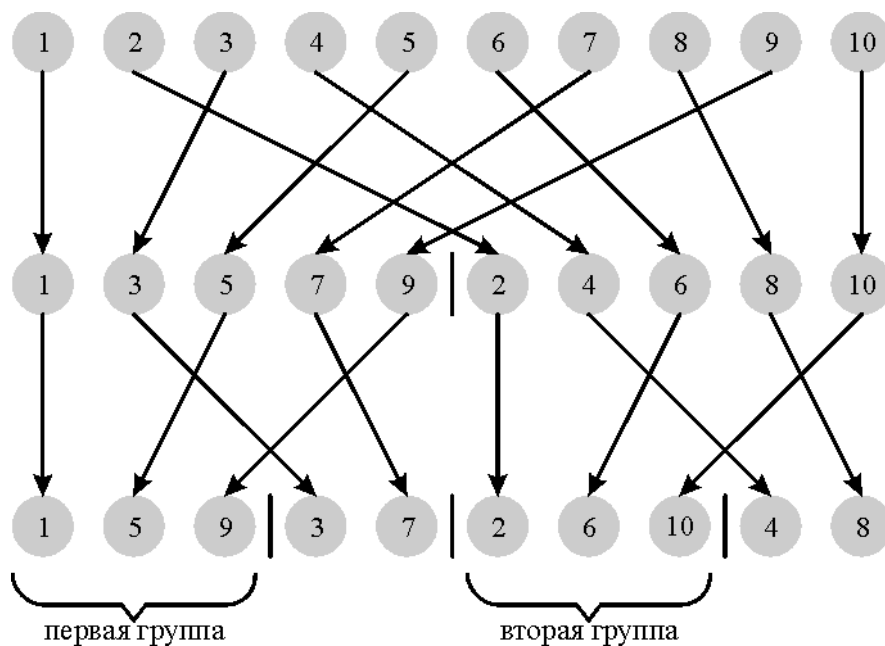


Рисунок 11 – Из условия задачи 4

Выходной файл должен содержать количество вариантов формирования групп разведки.

Ход решения задачи 4

Данный алгоритм решает задачу по вычислению количества вариантов формирования групп разведки. Программа получает количество солдат и делит его на группы согласно описанному в задаче условию, после чего подсчитывает количество групп из трех человек, которые возможно сформировать.

```
using System;
```

```
class Program {  
    static int Gruppi(int a) {  
        if(a == 3) return 1;  
        else if(a > 3) {  
            return Gruppi(a/2) + Gruppi(a-a/2);  
        }  
        else return 0;  
    }  
    public static void Main(string[] args) {  
        int n = Convert.ToInt32(Console.ReadLine());  
        Console.WriteLine(Gruppi(n));  
    }  
}
```



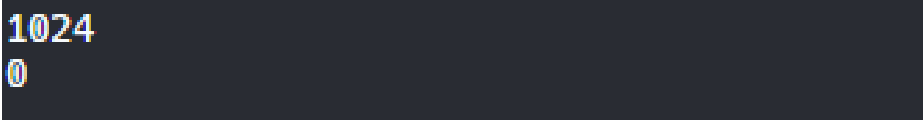
```
10  
2
```

Рисунок 12 – Первый тест задачи 4



```
4  
0
```

Рисунок 13 – Второй тест задачи 4



```
1024  
0
```

Заключение

В ходе выполнения задач данной расчетно-графической работы были использованы различные средства языка программирования С#. Были рассмотрены методы, циклы, различные способы работы с переменными и структурами данных. Благодаря выполнению этой работы получилось значительно улучшить знание синтаксиса и в целом навыки программирования на С#, составления алгоритмов, использования различных приемов для решения задач по программированию высокого уровня сложности.

Литература

1. <https://learn.microsoft.com/ru-ru/dotnet/fundamentals/runtime-libraries/system-datetime#initialize-a-datetime-object> (дата обращения 12.12.2024)
2. <https://metanit.com/sharp/tutorial/19.1.php> (дата обращения 14.12.2024)
3. <https://proglib.io/p/samouchitel-po-c-dlya-nachinayushchih-chast-1-ustanovite-sredu-razrabotki-i-osvoyte-osnovy-yazyka-za-30-minut-2021-11-23> (дата обращения 01.12.2024)
4. <https://devpractice.ru/csharp-basic-getting-start/> (дата обращения 02.12.2024)
5. <https://dotnet.microsoft.com/ru-ru/languages/csharp> (дата обращения 03.12.2024)