Упражнения: Повторения с цикли

Тествайте решенията си в **Judge системата**: https://judge.softuni.bg/Contests/3157/Loops

1. Числа от 1 до 100

Напишете програма, която отпечатва числата от 1 до 100, всяко на нов ред.

Примерен вход и изход

Вход	Изход
(няма)	1 2 3 … 98 99 100

Насоки

1. Създайте **нов проект с име** "**Numbers1To100**".
2. Отидете в тялото на метода **Main(String[] args)** и напишете решението на задачата. Можете да си помогнете с кода от картинката по-долу:

```
static void Main(string[] args)
{
    for (int i = 1; i <= 100; i++)
    {
        Console.WriteLine(i);
    }
}
```

2. Числата от N до 1 в обратен ред

Напишете програма, която чете цяло положително число **n**, въведено от потребителя, и печата **числата от n до 1 в обратен ред** (от най-голямото към най-малкото).

Примерен вход и изход

Вход	Изход
2	2 1

Вход	Изход
3	3 2 1

Вход	Изход
5	5 4 3 2 1

Насоки

1. Създайте **нов проект с име** "**NumbersNTo1**".

2. Отидете в тялото на метода **`Main(String[] args)`** и напишете решението на задачата. Можете да си помогнете с кода от картинката по-долу и обърнете внимание, че се използва цикъл с отрицателна стъпка:

```
static void Main(string[] args)
{
    int n = int.Parse(Console.ReadLine());
    for (int i = n; i >= 1; i--)
    {
        Console.WriteLine(i);
    }
}
```

3. Числата от 1 до N през 3

Напишете програма, която чете число **n**, въведено от потребителя, и отпечатва **числата от 1 до n през 3** (със стъпка 3).

Примерен вход и изход

Вход	Изход
10	1 4 7 10

Вход	Изход
7	1 4 7

Вход	Изход
15	1 4 7 10 13

Насоки

1. Създайте **нов проект с име** "**`Number1ToNWithStep3`**".
2. Отидете в тялото на метода **`Main(String[] args)`** и напишете решението на задачата. Можете да си помогнете с кода от картинката по-долу и обърнете внимание на цикъла, че е със стъпка 3:

```
static void Main(string[] args)
{
    int n = int.Parse(Console.ReadLine());
    for (int i = 1; i <= n; i += 3)
    {
        Console.WriteLine(i);
    }
}
```

4. Четни степени на 2

Да се напише програма, която чете число **n**, въведено от потребителя, и **печата четните степени на 2 ≤ 2^n**: **2^0**, **2^2**, **2^4**, **2^6**, …, **2^n**.

Примерен вход и изход

Вход	Изход
3	1 4

Вход	Изход
4	1 4 16

Вход	Изход
5	1 4 16

Вход	Изход
6	1 4 16 64

Вход	Изход
7	1 4 16 64

5. Поток от символи

Напишете програма, която чете текст(стринг), въведен от потребителя и печата всеки символ от текста на отделен ред.

Примерен вход и изход

Вход	Изход	Вход	Изход
softuni	s o f t u n i	ice cream	i c e c r e a m

Насоки

1. Прочетете входният текст:

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

2. Направете **for** цикъл с начална стойност на контролната променлива от **0** до **input.Length** (дължината на текста). На всяка итерация взимайте буквата на позиция във въведената дума равна на стойността на контролната променлива **i**, чрез метода **charAt()**

```
for (int i = 0; i < input.Length; i++)
{
    char letter = input[i];
}
```

3. На всяка итерация принтирайте стойността на променливата **letter**:

```
for (int i = 0; i < input.Length; i++)
{
    char letter = input[i];
    Console.WriteLine(letter);
}
```

6. Сумиране на гласните букви

Да се напише програма, която чете **текст** (стринг), въведен от потребителя, и изчислява и отпечатва **сумата от стойностите на гласните букви** според таблицата по-долу:

буква	a	e	i	o	u
стойност	1	2	3	4	5

Примерен вход и изход

Вход	Изход	Коментар
hello	6	e + o = 2 + 4 = 6

hi	3	i = 3
bamboo	9	a+o+o = 1+4+4 = 9

7. Сумиране на числа

Да се напише програма, която **чете n-на** брой **цели числа**, въведени от потребителя **и ги сумира**.

- От първия ред на входа се въвежда броят числа **n**.
- От следващите **n** реда се въвежда по едно цяло число.

Програмата трябва да прочете числата, да ги сумира и да отпечата сумата им.

Примерен вход и изход

Вход	Изход
2 10 20	30

Вход	Изход
3 -10 -20 -30	-60

Вход	Изход
4 45 -20 7 11	43

Вход	Изход
1 999	999

Вход	Изход
0	0

8. Редица цели числа

Напишете програма, която чете **n на брой цели числа**. Принтирайте **най-голямото** и **най-малкото** число сред въведените.

Примерен вход и изход

Вход	Изход
5 10 20 304 0 50	Max number: 304 Min number: 0

Вход	Изход
6 250 5 2 0 100 1000	Max number: 1000 Min number: 0

9. Лява и дясна сума

Да се напише програма, която чете **2*n-на брой** цели числа, подадени от потребителя, и проверява дали **сумата на първите n числа** (лява сума) е равна на **сумата на вторите n числа** (дясна сума). При равенство печата "**Yes, sum =** " + **сумата**; иначе печата "**No, diff =** " + **разликата**. Разликата се изчислява като положително число (по абсолютна стойност).

Примерен вход и изход

Вход	Изход	Коментар
2 10 90 60 40	Yes, sum = 100	10+90 = 60+40 = 100

Вход	Изход	Коментар
2 90 9 50 50	No, diff = 1	90+9 ≠ 50+50 Difference = |99-100| = 1

10. Четна / нечетна сума

Да се напише програма, която чете **n-на брой** цели числа, подадени от потребителя, и проверява дали **сумата от числата на четни позиции** е равна на **сумата на числата на нечетни позиции**. При равенство да се отпечатат два реда: "**Yes**" и на нов ред "**Sum =** " + **сумата**; иначе да се отпечата "**No**" и на нов ред "**Diff =** " + **разликата**. Разликата се изчислява по абсолютна стойност.

Примерен вход и изход

Вход	Изход	Коментар
4 10 50 60 20	Yes Sum = 70	10+60 = 50+20 = 70

Вход	Изход	Коментар
4 3 5 1 -2	No Diff = 1	3+1 ≠ 5-2 Diff = |4-3| = 1

Вход	Изход	Коментар
3 5 8 1	No Diff = 2	5+1 ≠ 8 Diff = |6-8| = 2

11. *Умната Лили

Лили вече е на **N години**. За всеки свой **рожден ден** тя получава подарък. За **нечетните** рождени дни (**1, 3, 5...n**) получава **играчки**, а за всеки **четен** (**2, 4, 6...n**) получава **пари**. За **втория рожден ден** получава **10.00 лв**, като **сумата се увеличава с 10.00 лв., за всеки следващ четен рожден ден** (**2 -> 10**, **4 -> 20**, **6 -> 30**...и т.н.). През годините Лили тайно е спестявала парите. **Братът** на Лили, **в годините**, които тя **получава пари**, **взима по 1.00 лев** от тях. Лили **продала играчките** получени през годините, **всяка за P лева** и **добавила** сумата **към спестените пари**. С парите искала да си **купи пералня за X лева**. Напишете програма, която да пресмята, **колко пари е събрала** и дали ѝ **стигат да си купи пералня**.

Вход

Програмата прочита **3 числа**, въведени от потребителя, на отделни редове:

- **Възрастта** на Лили - **цяло число** в интервала **[1...77]**
- **Цената на пералнята** – реално число
- **Цена на играчки** – реално число

Изход

Да се отпечата на конзолата един ред:

- Ако парите на Лили са достатъчни:
  - **"Yes! {N}"** - където **N** е остатъка пари след покупката
- Ако парите не са достатъчни:
  - **"No! {M}"** - където **M** е сумата, която не достига
- Числата **N** и **M** трябва да за **форматирани до вторият знак след десетичната запетая**.

Примерен вход и изход

Вход	Изход	Коментар
10 170.00 6	Yes! 5.00	**Първи рожден ден** получава **играчка**; **2ри -> 10лв**; 3ти -> играчка; **4ти** -> 10 + 10 = **20лв**; **5ти** -> играчка; **6ти** -> 20 + 10 = **30лв**; **7ми** -> играчка; **8ми** -> 30 + 10 = **40лв**; **9ти** -> играчка; **10ти** -> 40 + 10 = **50лв**. **Спестила е** -> 10 + 20 + 30 + 40 + 50 = **150лв**. Продала е **5 играчки по 6 лв.** = **30лв**. **Брат ѝ взел 5 пъти по 1 лев** = **5лв**. **Остават** -> 150 + 30 – 5 = **175лв**.

		175 >= 170 (цената на пералнята) **успяла** е да я купи и са и **останали** 175-170 = **5 лв.**
21 1570.98 3	No! 997.98	**Спестила е 550лв**. **Продала** е **11** играчки по **3 лв.** = **33лв**. Брат ѝ **взимал 10** години по **1 лев** = **10лв**. **Останали** 550 + 33 – 10 = **573лв** **573 < 1570.98** – **не е успяла** да купи пералня. **Не ѝ достигат** 1570.98–573 = **997.98лв**

12. Четене на думи

Напишете програма, която чете текст от конзолата(стринг) и го принтира, докато не получи командата "**Stop**".

Примерен вход и изход

Вход	Изход
Nakov SoftUni Sofia Bulgaria SomeText Stop AfterStop Europe HelloWorld	Nakov SoftUni Sofia Bulgaria SomeText

Вход	Изход
Sofia Berlin Moscow Athens Madrid London Paris Stop AfterStop	Sofia Berlin Moscow Athens Madrid London Paris

13. Парола

Напишете програма, която първоначално прочита име и парола на потребителски профил. След това чете парола за вход, при въвеждане на грешна парола, потребителя да се подкани да въведе нова парола.

Примерен вход и изход

Вход	Изход
Nakov 1234 pass 1324 1234	Welcome Nakov!

Вход	Изход
Gosho secret secret	Welcome Gosho!

Насоки

1. Инициализирайте две променливи **`username`** и **`password`**, които ще съдържат потребителското име и паролата:

```
string username = Console.ReadLine();
string password = Console.ReadLine();
```

2. Инициализирайте променлива **`input`**, която ще държи въведената от потребителя парола за вход:

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

3. В **`while`** цикъл, до въвеждане на валидна парола, четете нова:

```
while (input != password)
{
    input = Console.ReadLine();
}
```

4. Когато се въведе **валидна парола принтирайте съобщението за успешен вход**:

```
string username = Console.ReadLine();
string password = Console.ReadLine();

string input = Console.ReadLine();
while (input != password)
{
    input = Console.ReadLine();
}

Console.WriteLine($"Welcome: {username}!");
```

14. Сума от числа

Напишете програма, която чете цяло число от конзолата и на всеки следващ ред цели числа, докато тяхната сума стане по-голяма или равна на първоначалното число.. След приключване да се отпечата **сумата на въведените числа**.

Примерен вход и изход

Вход	Изход
100 10 20 30 40	100

Вход	Изход
20 1 2 3 4 5 6	21

15. Редица числа 2K+1

Напишете програма, която чете число **n**, въведено от потребителя, и отпечатва **всички числа ≤ n от редицата**: 1, 3, 7, 15, 31, …. Всяко следващо число се изчислява като умножим **предишното** с **2** и добавим **1**.

Примерен вход и изход

Вход	Изход
3	1 3

Вход	Изход
8	1 3 7

Вход	Изход
17	1 3 7 15

Вход	Изход
31	1 3 7 15 31

Насоки

1. Прочетете от конзолата цяло число.
```
int number = int.Parse(Console.ReadLine());
```
2. Създайте променлива от тип **цяло число**, която ще е брояч и има **първоначална стойност 1**.
```
int k = 1;
```
3. Създайте **while** цикъл, който се повтаря докато **брояча е по-малък** или равен на числото, което сте прочели от конзолата.
```
while(k <= num)
{

}
```
4. При всяко повторение на цикъла **принтирайте стойността на брояча** и му **прибавяйте дадената стойност**.
```
while(k <= num)
{
    Console.WriteLine(k);
    k = k * 2 + 1;
}
```

16. Баланс по сметка

Напишете програма, която пресмята колко общо пари има в сметката, след като направите определен брой вноски. На всеки ред ще получавате сумата, която трябва да внесете в сметката, **до получаване на команда "NoMoreMoney"** . При всяка получена сума на конзолата трябва да се извежда **"Increase: "** + сумата и тя да се **прибавя в сметката**. Ако получите число **по-малко от 0** на конзолата трябва да се изведе **"Invalid operation!"** и **програмата да приключи**. Когато програмата приключи трябва да се принтира **"Total: "** + общата сума в сметката закръглена до втория знак след десетичната запетая.

Примерен вход и изход

Вход	Изход
5.51 69.42 100 NoMoreMoney	Increase: 5.51 Increase: 69.42 Increase: 100 Total: 174.93

Вход	Изход
120 45.55 -150	Increase: 120 Increase: 45.55 Invalid operation! Total: 165.55

17. Най-голямо число

Напишете програма, която до получаване на **командата "Stop",** чете **цели числа**, въведени от потребителя, и намира **най-голямото** измежду тях. Въвежда се по едно число на ред.

Примерен вход и изход

Вход	Изход
100 99 80	100

Вход	Изход
-10 20 -30	20

Вход	Изход
45 -20 7	99

Вход	Изход
999 Stop	999

Вход	Изход
-1 -2 Stop	-1

70 Stop		Stop		99 Stop					

18. Най-малко число

Напишете програма, която до получаване на **командата "Stop",** чете **цели числа**, въведени от потребителя, и намира **най-малкото** измежду тях. Въвежда се по едно число на ред.

Примерен вход и изход

Вход	Изход
100 99 80 70 Stop	70

Вход	Изход
-10 20 -30 Stop	-30

Вход	Изход
45 -20 7 99 Stop	-20

Вход	Изход
999 Stop	999

Вход	Изход
-1 -2 Stop	-2

19. Завършване

Напишете програма, която изчислява **средната оценка** на ученик от цялото му обучение. На първия ред ще получите **името на ученика**, а на всеки следващ ред неговите годишни оценки. Ученикът преминава в следващия клас, **ако годишната му оценка е по-голяма или равна на 4.00**. **Ако ученикът бъде скъсан повече от един път, то той бива изключен и програмата приключва,** като се отпечатва **името на ученика и в кой клас бива изключен**.

При успешно завършване на **12-ти** клас да се отпечата :

"**{име на ученика} graduated. Average grade: {средната оценка от цялото обучение}**"

В случай, че ученикът е изключен от училище, да се отпечата:

"**{име на ученика} has been excluded at {класа, в който е бил изключен} grade**"

Стойността трябва да бъде форматирана до втория знак след десетичната запетая.

Примерен вход и изход

Вход	Изход
Gosho 5 5.5 6 5.43 5.5 6 5.55 5 6 6 5.43 5	Gosho graduated. Average grade: 5.53

Вход	Изход
Mimi 5 6 5 6 5 6 6 2 3	Mimi has been excluded at 8 grade

20. *Преместване

На осемнадесетия си рожден ден на Хосе взел решение, че ще се изнесе да живее на квартира. Опаковал багажа си в **кашони** и намерил подходяща обява за апартамент под наем. Той започва да пренася своя багаж

на части, защото не може да пренесе целия наведнъж. Има ограничено **свободно пространство** в новото си жилище, където може да разположи вещите, така че мястото да бъде подходящо за живеене.

Напишете **програма, която изчислява свободния обем от жилището на Хосе, който остава след като пренесе багажа си.**

Бележка: Един кашон е с точни размери: 1m. x 1m. x 1m.

Вход

Потребителят въвежда следните данни на отделни редове:

- **Широчина на свободното пространство - цяло число в интервала [1...1000]**
- **Дължина на свободното пространство - цяло число в интервала [1...1000]**
- **Височина на свободното пространство - цяло число в интервала [1...1000]**
- На следващите редове **(до получаване на команда "Done") - брой кашони, които се пренасят в квартирата - цели числа в интервала [1...10000];**

Програмата трябва да приключи прочитането на данни при команда "Done" или ако свободното място свърши.

Изход

Да се **отпечата** на конзолата **един** от следните редове:

- Ако стигнете до командата **"Done"** и има още свободно място:
  "{брой свободни куб. метри} Cubic meters left."
- Ако свободното място свърши преди да е дошла команда **"Done":**
  "No more free space! You need {брой недостигащи куб. метри} Cubic meters more."

Примерен вход и изход

Вход	Изход	Обяснение
10 10 2 20 20 20 20 122	No more free space! You need 2 Cubic meters more.	10 * 10 * 2 = **200 кубични метра.** 20 + 20 + 20 + 20 + 122 = **202 кубични метра.** 200 - 202 = **2 недостигащи** кубични метра
10 1 2 4 6 **Done**	10 Cubic meters left.	10 * 1 * 2 = **20 кубични метра.** 4 + 6 = **10 кубични метра.** 20 - 10 = **10 кубични метра.**

21. Часовник

Напишете програма, която отпечатва **часовете в денонощието от 0:0 до 23:59**, всеки на отделен ред. Часовете трябва да се изписват във формат **"{час}:{минути}"**.

Примерен вход и изход

Вход	Изход

(няма вход)	0:0 0:1 0:2 0:3 0:4 0:5 0:6 0:7 0:8 0:9 0:10 ... 23:50 23:51 23:52 23:53 23:54 23:55 23:56 23:57 23:58 23:59

Насоки

1. Създайте 2 вложени for-цикъла, с които да итерирате през всяка една минута и час от денонощието:

```
for (int h = 0; h <= 23; h++)
{
    for (int m = 0; m <= 59; m++)
    {

    }
}
```

2. Отпечатайте резултата:

```
for (int h = 0; h <= 23; h++)
{
    for (int m = 0; m <= 59; m++)
    {
        Console.WriteLine($"{h}:{m}");
    }
}
```

22. Таблица за умножение

Отпечатайте на конзолата таблицата за умножение за числата от 1 до 10 във формат:
"{първи множител} * {втори множител} = {резултат}".

Примерен вход и изход

Вход	Изход
(няма вход)	1 * 1 = 1 1 * 2 = 2

	1 * 3 = 3 1 * 4 = 4 1 * 5 = 5 1 * 6 = 6 1 * 7 = 7 1 * 8 = 8 1 * 9 = 9 1 * 10 = 10 ... 10 * 1 = 10 10 * 2 = 20 10 * 3 = 30 10 * 4 = 40 10 * 5 = 50 10 * 6 = 60 10 * 7 = 70 10 * 8 = 80 10 * 9 = 90 10 * 10 = 100

Насоки

3. Създайте 2 вложени for-цикъла, с които да итерирате всяка възможна стойност на двата множителя от 1 до 10:

```
for (int x = 1; x <= 10; x++)
{
    for (int y = 1; y <= 10; y++)
    {

    }
}
```

4. Намерете произведението на двата множителя и отпечатайте резултата:

```
for (int x = 1; x <= 10; x++)
{
    for (int y = 1; y <= 10; y++)
    {
        int product = x * y;
        Console.WriteLine($"{x} * {y} = {product}");
    }
}
```

23. Комбинации

Напишете програма, която изчислява **колко решения в естествените числа** (включително и нулата) има уравнението:

x1 + x2 + x3 = n

Числото n е цяло число и се въвежда от конзолата.

Примерен вход и изход

Вход	Изход	Обяснения	Вход	Изход	Вход	Изход

25	351	Генерираме всички комбинации от 3 числа, като първата е: 0+0+0=0, но понеже не е равна на 25, продължаваме: 0+0+1=1 – също не е 25 и т.н. Стигаме до първата валидна комбинация: 0 + 0 + 25 = 25, увеличаваме броя на валидни комбинации с 1,втората валидна комбинация е: 0 + 1 + 24 = 25 Третата: 0 + 2 + 23 = 25 и т.н. След генериране на всички възможни комбинации, броят на валидните е 351.	20	231	5	21

Насоки

1. Прочетете входните данни – едно цяло число, въведено от потребителя и го запаметете в променлива:

```
int n = int.Parse(Console.ReadLine());
```

2. Създайте 3 вложени for-цикъла, с които да итерирате всяка възможна стойност на едно от 3те числа в уравнението:

```
int n = int.Parse(Console.ReadLine());

for (int x1 = 0; x1 <= n; x1++)
{
    for (int x2 = 0; x2 <= n; x2++)
    {
        for (int x3 = 0; x3 <= n; x3++)
        {

        }
    }
}
```

3. Направете проверка в най-вътрешния вложен цикъл за стойностите на **x1**, **x2**, **x3** във всяка една итерация. За да бъде валидно уравнението, техният сбор трябва да е равен на **n**. Създайте променлива **validCombinationsCount**, която да пази броя на валидните комбинации и добавяйте към нея всеки път, когато генерирате такава:

```
int n = int.Parse(Console.ReadLine());

for (int x1 = 0; x1 <= n; x1++)
{
    for (int x2 = 0; x2 <= n; x2++)
    {
        for (int x3 = 0; x3 <= n; x3++)
        {
            if (x1 + x2 + x3 == n)
            {
                // TODO: Add to valid combination counter
            }
        }
    }
}
```

4. Накрая принтирайте броя на валидните комбинации (**validCombinationsCount**).

24. Сума от две числа

Напишете програма която проверява **всички възможни комбинации от двойка числа в интервала от две дадени числа**. На изхода се отпечатва, **коя поред е комбинацията** чиито **сбор от числата е равен** на дадено **магическо число**. Ако няма **нито една комбинация** отговаряща на условието се отпечатва **съобщение, че не е намерено**.

Вход

Входът се чете от конзолата и се състои от **три реда**:

- **Първи ред** – **начало на интервала** – **цяло число** в интервала **[1...999]**
- **Втори ред** – **край на интервала** – **цяло число** в интервала **[по-голямо от първото число...1000]**
- **Трети ред – магическото число – цяло число** в интервала **[1...10000]**

Изход

На конзолата трябва да се отпечата **един ред,** според резултата:

- Ако **е намерена комбинация** чиито **сбор на числата е равен на магическото число**
  - **"Combination N:{пореден номер} ({първото число} + {второ число} = {магическото число})"**
- Ако **не е намерена** комбинация отговаряща на условието
  - **"{броят на всички комбинации} combinations - neither equals {магическото число}"**

Примерен вход и изход

Вход	Изход	Обяснения	Вход	Изход
1 10 5	Combination N:4 (1 + 4 = 5)	Всички комбинации от две числа между 1 и 10 са: 1 1, 1 2, 1 3, **1 4**, 1 5, ... 2 1, 2 2, ... 4 9, 4 10, 5 1 ... 10 9, 10 10 Първата комбинация, чиито сбор на числата е равен на магическото число 5 е **четвъртата (1 и 4)**	88 888 1000	Combination N:20025 (112 + 888 = 1000)
Вход	**Изход**	**Обяснения**	**Вход**	**Изход**
23 24 20	4 combinations - neither equals 20	Всички комбинации от две числа между 23 и 24 са: 23 23, 23 24, 24 23, 24 24 (общо 4) Няма двойки числа, чиито сбор е равен на магическото 20	88 888 2000	641601 combinations - neither equals 2000

25. Пътуване

Ани обича да пътува и иска тази година да посети **няколко** различни дестинации. Като си избере дестинация, ще прецени **колко пари ще й трябват**, за да отиде до там и ще започне да **спестява**. Когато е спестила **достатъчно**, ще може да пътува.

От **конзолата всеки път ще се четат първо дестинацията и минималния бюджет**, който ще е нужен за пътуването.

След това ще се четат **няколко суми**, които Ани спестява като работи и **когато успее да събере достатъчно за пътуването, ще заминава, като на конзолата трябва да се изпише:**

"Going to {дестинацията}!"

Когато е посетила всички дестинации, които иска, **вместо дестинация ще въведе "End"** и програмата ще приключи.

Примерен вход и изход

Вход	Изход	Вход	Изход
Greece 1000 200 200 300 100 150 240 Spain 1200 300 500 193 423 End	Going to Greece! Going to Spain!	France 2000 300 300 200 400 190 258 360 Portugal 1450 400 400 200 300 300 Egypt 1900 1000 280 300 500 End	Going to France! Going to Portugal! Going to Egypt!

26. Сграда

Напишете програма, която извежда на конзолата номерата на стаите в една сграда (в низходящ ред), като са изпълнени следните условия:

- На **всеки четен етаж има само офиси**
- На **всеки нечетен етаж има само апартаменти**
- Всеки **апартамент** се означава по следния начин : **A{номер на етажа}{номер на апартамента}**, **номерата на апартаментите започват от 0.**
- Всеки **офис** се означава по следния начин : **O{номер на етажа}{номер на офиса}**, **номерата на офисите също започват от 0.**
- **На последният етаж винаги има апартаменти** и те са по-големи от останалите, за това **пред номера им пише 'L', вместо 'A'. Ако има само един етаж, то има само големи апартаменти!**

От конзолата се прочитат две **цели числа - броят на етажите и броят на стаите за един етаж.**

Примерен вход и изход

Вход	Изход	Обяснения

6 4	L60 L61 L62 L63 A50 A51 A52 A53 O40 O41 O42 O43 A30 A31 A32 A33 O20 O21 O22 O23 A10 A11 A12 A13	Имаме общо 6 етажа, с по 4 стаи на етаж. Нечетните етажи имат само апартаменти, а четните само офиси.

Вход	Изход	Вход	Изход
9 5	L90 L91 L92 L93 L94 O80 O81 O82 O83 O84 A70 A71 A72 A73 A74 O60 O61 O62 O63 O64 A50 A51 A52 A53 A54 O40 O41 O42 O43 O44 A30 A31 A32 A33 A34 O20 O21 O22 O23 O24 A10 A11 A12 A13 A14	4 4	L40 L41 L42 L43 A30 A31 A32 A33 O20 O21 O22 O23 A10 A11 A12 A13

27. * Билети за кино

Вашата задача е да **напишете програма**, която да изчислява **процента на билетите за всеки тип от продадените билети**: студентски(**student**), стандартен(**standard**) и детски(**kid**), за всички прожекции. Трябва да изчислите и **колко процента от залата е запълнен**а за **всяка една прожекция.**

Вход

Входът е поредица от **цели числа** и **текст**:

- На първия ред до получаване на командата "**Finish**" - име на филма – **текст**
- На втори ред – свободните места в салона за всяка прожекция – **цяло число [1 … 100]**
- За всеки филм, се чете по един ред до изчерпване на свободните места в залата или до получаване на командата "**End**":
  - Типа на закупения билет - текст ("**student", "standard", "kid**")

Изход

На конзолата трябва да се печатат **следните редове**:

- След всеки филм да се отпечата, колко процента от кино залата е пълна
  "{името на филма} - {процент запълненост на залата}% full."
- При получаване на командата "Finish" да се отпечатат четири реда:
  - **"Total tickets: {общият брой закупени билети за всички филми}"**
  - **"{процент на студентските билети}% student tickets."**
  - **"{процент на стандартните билети}% standard tickets."**
  - **"{процент на детските билети}% kids tickets."**

Примерен вход и изход

Вход	Изход	Обяснения
Taxi 10 standard kid student student standard standard	Taxi - 60.00% full. Scary Movie - 100.00% full. Total tickets: 12 66.67% student tickets. 25.00% standard tickets. 8.33% kids tickets.	Първи филм – Taxi, местата в залата са 10 Купуват се 3 стандарти, 2 студентски, 1 детски билет и получаваме командата End. Общо 6 билета от 10 места -> 60% от залата е заета. Втори филм – Scary Movie, места в залата са 6 Купуват се 6 студентски билета и местата в залата свършват.

End Scary Movie 6 student student student student student student Finish		Общо 6 билета от 6 места -> 100% от залата е заета. Получаваме командата Finish Общо закупените билети за всички филми са 12. За всички филми са закупени общо: 8 студентски билета. 8 билета от общо 12 е 66.67% 3 стандартни билета. 3 билета от общо 12 е 25% 1 детски билет. 1 билет от общо 12 е 8.33%

Вход	Изход	Обяснения
The Matrix 20 student standard kid kid student student standard student End The Green Mile 17 student standard standard student standard student End Amadeus 3 standard standard standard Finish	The Matrix - 40.00% full. The Green Mile - 35.29% full. Amadeus - 100.00% full. Total tickets: 17 41.18% student tickets. 47.06% standard tickets. 11.76% kids tickets.	Първи филм – The Matrix, местата в залата са 20 Купуват се 2 стандартни, 4 студентски, 2 детски билета и получаваме командата End. Общо 8 билета от 20 места -> 41.18% от залата е заета Втори филм - The Green Mile, местата в залата са 17 Купуват се 3 стандартни, 3 студентски билета и получаваме командата End. Общо 6 билета от 17 места -> 47.06% от залата е заета Трети филм – Amadeus, местата в залата са 3 Купуват се 3 стандартни билета и местата в залата свършват. Общо 3 билета от 3 места -> 100% от залата е заета. Получаваме командата Finish Общо закупените билети за всички филми са 17. За всички филми са закупени общо: 7 студентски билета. 7 билета от общо 17 е 41.18% 8 стандартни билета. 8 билета от общо 17 е 47.06% 2 детски билета. 2 билета от общо 17 е 11.76%

28. Числа до 1000, завършващи на 7

Напишете програма, която отпечатва числата в диапазона **[1…1000]**, които **завършват на 7**.

Вход	Изход
(няма)	7 17 27 … 997

Насоки

1. Направете **for цикъл** от 7 до 997 и проверете всяко число дали завършва на 7. Едно число завършва на 7, когато резултатът от **модулното деление на числото и 10 е равен на 7**.

```
for (int i = 7; i <= 997; i++)
{
    if (i % 10 == 7)
    {
        Console.WriteLine(i);
    }
}
```

29. Елемент, равен на сумата на останалите

Да се напише програма, която чете **n-на брой** цели числа, въведени от потребителя, и проверява дали сред тях съществува число, което е равно на сумата на всички останали. Ако има такъв елемент, печата "**Yes**", "**Sum = " + неговата стойност**; иначе печата "**No**", "**Diff = " + разликата между най-големия елемент и сумата на останалите** (по абсолютна стойност).

Примерен вход и изход

Вход	Изход	Коментар
7 3 4 1 1 2 **12** 1	Yes Sum = 12	3 + 4 + 1 + 2 + 1 + 1 = 12
4 **6** 1 2 3	Yes Sum = 6	1 + 2 + 3 = 12
3 1 1 10	No Diff = 8	|10 - (1 + 1)| = 8
3 5 5 1	No Diff = 1	|5 - (5 + 1)| = 1
3 1 1 1	No Diff = 1	

Насоки

1. Прочетете число **n** и завъртете цикъл до него, като на всеки нов ред четете число **num**.

```
int n = int.Parse(Console.ReadLine());
for (int i = 0; i < n; i++)
{
    int num = int.Parse(Console.ReadLine());
}
```

2. Създайте променливи **sum**, която ще държи сумата на **num** и **max**, която ще пази най-голямото число.

```
int sum = 0;
int max = int.MinValue;
for (int i = 0; i < n; i++)
{
    int num = int.Parse(Console.ReadLine());
    sum += num;

    if (num > max)
    {
        max = num;
    }
}
```

3. Проверете дали сумата е равна на **max** и принтирайте съответният изход.

```
int sumWithoutMaxNumber = sum - max;
if (max == sumWithoutMaxNumber)
{
    Console.WriteLine("Yes");
    Console.WriteLine("Sum = " + max);
}
else
{
    int diff = Math.Abs(max - sumWithoutMaxNumber);
    Console.WriteLine("No");
    Console.WriteLine("Diff = " + diff);
}
```

30. Четни / нечетни позиции

Напишете програма, която чете **n-на брой числа**, въведени от потребителя, и пресмята **сумата**, **минимума** и **максимума** на числата на **четни** и **нечетни** позиции (броим от 1). Когато няма минимален / максимален елемент, отпечатайте "**No**".

Изходът да се форматира в следния вид:

"**OddSum=**" + {**сума** на числата на **нечетни** позиции},

"**OddMin=**" + { **минимална** стойност на числата на **нечетни** позиции } / {"**No**"},

"**OddMax=**" + { **максимална** стойност на числата на **нечетни** позиции } / {"**No**"},

"**EvenSum=**" + { **сума** на числата на **четни** позиции },

"**EvenMin=**" + { **минимална** стойност на числата на **четни** позиции } / {"**No**"},

"**EvenMax=**" + { **максимална** стойност на числата на **четни** позиции } / {"**No**"}

Всяко число трябва да е форматирано до втория знак след десетичната запетая.

Примерен вход и изход

Вход	Изход
6 **2** 3 **5** 4 **2** 1	OddSum=9.00, OddMin=2.00, OddMax=5.00, EvenSum=8.00, EvenMin=1.00, EvenMax=4.00

Вход	Изход
2 **1.5** -2.5	OddSum=1.50, OddMin=1.50, OddMax=1.50, EvenSum=-2.50, EvenMin=-2.50, EvenMax=-2.50

Вход	Изход
1 **1**	OddSum=1.00, OddMin=1.00, OddMax=1.00, EvenSum=0.00, EvenMin=No, EvenMax=No

Вход	Изход
0	OddSum=0.00, OddMin=No, OddMax=No, EvenSum=0.00, EvenMin=No, EvenMax=No

Вход	Изход
5 **3** -2 **8** 11 **-3**	OddSum=8.00, OddMin=-3.00, OddMax=8.00, EvenSum=9.00, EvenMin=-2.00, EvenMax=11.00

Вход	Изход
4 1.5 **1.75** 1.5 **1.75**	OddSum=3.00, OddMin=1.50, OddMax=1.50, EvenSum=3.50, EvenMin=1.75, EvenMax=1.75

Вход	Изход
1 **-5**	OddSum=-5.00, OddMin=-5.00, OddMax=-5.00, EvenSum=0.00, EvenMin=No, EvenMax=No

Вход	Изход
3 **-1** -2 **-3**	OddSum=-4.00, OddMin=-3.00, OddMax=-1.00, EvenSum=-2.00, EvenMin=-2.00, EvenMax=-2.00

Задача обединява няколко предходни задачи: намиране на **минимум**, намиране на **максимум**, намиране на **сума** и обработка на елементите от **четни и нечетни позиции**. Припомнете си ги.

Насоки

1. Работете с **реални числа** (не цели). Сумата, минимумът и максимумът също са реални числа.
2. Използвайте **неутрална начална стойност** при намиране на минимум / максимум, например **1000000000.0** и **-1000000000.0**. Ако получите накрая неутралната стойност, печатайте "**No**".
3. Завъртете **`for`** цикъл до числото, което ви се въвежда като на всеки нов ред прочитате ново число **`num`**.
4. Проверете дали **позицията на числото** е **четна** или **нечетна**, като променливата инициализирана в цикъла **`i`** отговаря на **позицията на числото**.
5. Ако позицията на числото е **четно**, увеличете **сумата на четните числа** и проверете дали числото е **по-голямо** от **най-голямото четно**, и му презапишете стойността. Също така проверете дали числото е **по-малко** от **най-малкото четно число** и му презапишете стойността.
6. Аналогично направете същото и за **нечетните числа**.

31. Хистограма

Дадени са **n цели числа** в интервала [**1**…**1000**]. От тях някакъв процент **p1** са под 200, друг процент **p2** са от 200 до 399, друг процент **p3** са от 400 до 599, друг процент **p4** са от 600 до 799 и останалите **p5** процента са от 800 нагоре. Да се напише програма, която изчислява и отпечатва процентите **p1**, **p2**, **p3**, **p4** и **p5**.

Пример: имаме n = **20** числа: 53, 7, 56, 180, 450, 920, 12, 7, 150, 250, 680, 2, 600, 200, 800, 799, 199, 46, 128, 65. Получаваме следното разпределение и визуализация:

Диапазон	Числа в диапазона	Брой числа	Процент
< 200	53, 7, 56, 180, 12, 7, 150, 2, 199, 46, 128, 65	12	p1 = 12 / 20 * 100 = **60.00**%
200 … 399	250, 200	2	p2 = 2 / 20 * 100 = **10.00**%
400 … 599	450	1	p3 = 1 / 20 * 100 = **5.00**%
600 … 799	680, 600, 799	3	p4 = 3 / 20 * 100 = **15.00**%
≥ 800	920, 800	2	p5 = 2 / 20 * 100 = **10.00**%

Вход

На първия ред от входа стои цялото число **n** (1 ≤ **n** ≤ 1000) – брой числа. На следващите **n реда** стои **по едно цяло число** в интервала [**1**…**1000**] – числата върху които да бъде изчислена хистограмата.

Изход

Да се отпечата на конзолата **хистограмата** – **5 реда**, всеки от които съдържа число между 0% и 100%, с точност две цифри след десетичната точка, например 25.00%, 66.67%, 57.14%.

Примерен вход и изход

Вход	Изход
3 1 2 999	66.67% 0.00% 0.00% 0.00% 33.33%

Вход	Изход
4 53 7 56 999	75.00% 0.00% 0.00% 0.00% 25.00%

Вход	Изход
7 800 801 250 199 399	14.29% 28.57% 14.29% 14.29% 28.57%

Вход	Изход
9 367 99 200 799 999	33.33% 33.33% 11.11% 11.11% 11.11%

Вход	Изход
14 53 7 56 180 450	57.14% 14.29% 7.14% 14.29% 7.14%

				599 799		333 555 111 9		920 12 7 150 250 680 2 600 200	

32. Деление без остатък

Дадени са **n-на брой цели числа** в интервала [**1**…**1000**]. От тях някакъв **процент p1 се делят без остатък на 2**, друг **процент p2** се **делят без остатък на 3**, друг **процент p3** се **делят без остатък на 4**. Да се напише програма, която изчислява и отпечатва процентите **p1**, **p2** и **p3**.

Пример: имаме n = **10** числа: 680, 2, 600, 200, 800, 799, 199, 46, 128, 65. Получаваме следното разпределение и визуализация:

Деление без остатък на:	**Числа в диапазона**	**Брой числа**	**Процент**
2	`680, 2, 600, 200, 800, 46, 128`	7	`p1 = 7.0 / 10 * 100 = 70.00%`
3	`600`	1	`p2 = 1 / 10 * 100 = 10.00%`
4	`680, 600, 200, 800, 128`	5	`p3 = 5 / 10 * 100 = 50.00%`

Вход

На първия ред от входа стои цялото число **n** (1 ≤ **n** ≤ 1000) - брой числа. На следващите **n реда** стои **по едно цяло число** в интервала [**1**…**1000**] - числата които да бъдат проверени на колко се делят.

Изход

Да се отпечатат на конзолата **3 реда**, всеки от които съдържа процент между 0% и 100%, с точност две цифри след десетичната точка, например 25.00%, 66.67%, 57.14%.

- На **първият ред** - процентът на числата които **се делят на 2**
- На **вторият ред** - процентът на числата които **се делят на 3**
- На **третият ред** - процентът на числата които **се делят на 4**

Примерен вход и изход

Вход	Изход
10 680 2 600 200 800 799 199 46	70.00% 10.00% 50.00%

Вход	Изход
3 3 6 9	33.33% 100.00% 0.00%

128 65				

33. Заплата

Шеф на компания забелязва че все повече служители прекарват време в сайтове, които ги разсейват. За да предотврати това, той въвежда изненадващи проверки на отворените табове на браузъра на служителите си. **Според сайта се налагат различни глоби:**

- "**Facebook**" -> **150 лв.**
- **"Instagram" -> 100 лв.**
- **"Reddit" -> 50 лв.**

От конзолата се четат два реда:

- **Брой отворени табове в браузъра n - цяло число в интервала [1...10]**
- **Заплата - число в интервала [500...1500]**

След това n – на брой пъти се чете име на уебсайт – текст

Ако по време на проверката заплатата стане **по-малка или равна на 0 лева**, на конзолата се изписва **"You have lost your salary." и програмата приключва.** В противен случай след проверката **на конзолата се изписва остатъкът** от заплатата (да се изпише **като цяло число**).

Примерен вход и изход

Вход	Изход	Обяснения
10 750 Facebook Dev.bg Instagram Facebook Reddit Facebook Facebook	You have lost your salary.	Има 10 отворени таба в браузъра. Заплатата е 750 За първия таб -> Facebook глоба 150 лв.(750 – 150 = 600) За втория таб -> Dev.bg не глобяват За третия таб -> Instagram глоба 100 лв.(600 – 100 = 500) За четвъртия таб -> Facebook глоба 150 лв.(500 – 150 = 350) За петия таб -> Reddit глоба 50 лв. (350 – 50 = 300) За шестия таб -> Facebook глоба 150 лв.(300 – 150 = 150) За седмия таб -> Facebook глоба 150 лв.(150 – 150 = 0) Заплатата е равна на 0, следователно се изписва съответният изход и програмата приключва.

Вход	Изход	Вход	Изход
3 500 Github.com Stackoverflow.com softuni.bg	500	3 500 Facebook Stackoverflow.com softuni.bg	350

34. Старата Библиотека

Ани отива до родния си град след много дълъг период извън страната. Прибирайки се вкъщи тя вижда старата библиотека на баба си и си спомня за любимата си книга. Помогнете на Ани, като напишете програма в която тя въвежда търсената от нея **книга**(**текст**). **Докато Ани не намери любимата си книга или не провери всички в библиотеката, програмата трябва да чете всеки път на нов ред името на всяка следваща книга (текст). Книгите в библиотеката са свършили щом получите текст "No More Books".**

- **Ако не открие търсената книгата да се отпечата на два реда:**
  - **"The book you search is not here!"**

- **"You checked {брой} books."**
- **Ако открие книгата си се отпечатва един ред:**
  - **"You checked {брой} books and found it."**

Примерен вход и изход

Вход	Изход	Обяснения
Troy Stronger Life Style Troy	You checked 2 books and found it.	Книгата която Ани търси, в случая е Troy. Първата книга от библиотеката е Stronger, втората е Life Style, третата книга е търсената – Troy и програмата приключва.
The Spot Hunger Games Harry Potter Torronto Spotify No More Books	The book you search is not here! You checked 4 books.	Книгата, която търси Ани е "The Spot". Първата книга от библиотеката е Hunger Games, втората Harry Potter, третата Torronto, а четвъртата Spotify. Понеже няма повече книги в библиотеката, четенето на имена приключва. Ани не намери книгата, която търсеше.
Bourne True Story Forever More Space The Girl Spaceship Strongest Profit Tripple Stella The Matrix Bourne	You checked 10 books and found it.	

Насоки

1. Прочетете входните данни от конзолата.

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

2. Направете **още две помощни променливи в началото**, които да следят **дали книгата е намерена или всички книги са проверени**. Едната променлива ще е **брояч** и трябва да е **от тип цяло число** и с **първоначална стойност нула**. С нея ще следим **колко книги са проверени**. Другата променлива трябва да е от **булев тип** и да е **с началната стойност false**.

```
int counter = 0;
bool isBookFound = false;
```

3. Направете си **while** цикъл, в който **всеки път** ще четете от конзолата нова книга, докато книгите в библиотеката **се изчерпят и прочетете текста "No More Books".**

```
string nextBookName = Console.ReadLine();
while (nextBookName != "No More Books")
{
    nextBookName = Console.ReadLine();
}
```

4. Ако книгата, която четете от конзолата **съвпада с любимата книга на Ани**, презапишете стойността на **променливата от булев тип**, и **прекратете цикъла**, в противен случай **увеличете брояча с едно**.

```
string nextBookName = Console.ReadLine();
while (nextBookName != "No More Books")
{
    if (nextBookName == favoriteBook)
    {
        isBookFound = true;
        break;
    }
    counter++;
    nextBookName = Console.ReadLine();
}
```

5. Според това, дали книгата е намерена, **принтирайте нужните съобщения**.

```
if (isBookFound)
{
    Console.WriteLine($"You checked {counter} books and found it.");
}
else
{
    Console.WriteLine("The book you search is not here!");
    Console.WriteLine($"You checked {counter} books.");
}
```

35. Подготовка за изпит

Напишете програма, в която Марин решава задачи от изпити **докато не получи** съобщение **"Enough"** от лектора си. При всяка решена задача той получава оценка. **Програмата трябва да приключи прочитането на данни при команда "Enough" или ако Марин получи определения брой незадоволителни оценки. Незадоволителна е всяка оценка, която е по-малка или равна на 4.**

Вход

- **На първи ред - брой незадоволителни оценки - цяло число в интервала [1…5]**
- **След това многократно се четат по два реда:**
  - **Име на задача - текст (низ)**
  - **Оценка - цяло число в интервала [2…6]**

Изход

- Ако Марин стигне до командата **"Enough", отпечатайте на 3 реда:**
  - **"Average score: {средна оценка}"**
  - **"Number of problems: {броя на всички задачи}"**
  - **"Last problem: {името на последната задача}"**
- **Ако получи определеният брой незадоволителни оценки:**
  - **"You need a break, {брой незадоволителни оценки} poor grades."**

Средната оценка да бъде форматирана до втория знак след десетичната запетая.

Примерен вход и изход

Вход	Изход	Обяснения

3 Money 6 Story 4 Spring Time 5 Bus 6 Enough	Average score: 5.25 Number of problems: 4 Last problem: Bus	Броя на позволени незадоволителни оценки е 3. Първата задача се казва Money, оценката на Марин е 6. Втората задача е Story, оценката на Марин е 4. Третата задача е Spring Time, оценката на Марин е 5. Четвъртата задача е Bus, оценката на Марин е 6. Следващата команда е Enough, програмата приключва. Средна оценка: 21 / 4 = 5.25 Брой решени задачи: 4 Последна задача: Bus
Вход	**Изход**	**Обяснения**
2 Income 3 Game Info 6 Best Player 4	You need a break, 2 poor grades.	Броят незадоволителни оценки е 2. Първата задача е Income, оценката на Марин е 3. Втората задача е Game Info, оценката на Марин е 6. Третата задача е Best Player, оценката на Марин е 4. Марин достигна допустимия брой незадоволителни оценки, време е за почивка.

Насоки

1. Прочетете входните данни от конзолата:

```
int failedThreshold = int.Parse(Console.ReadLine());
```

2. Направете **четири помощни променливи** в началото:
   - **брояч за незадоволителни оценки - с първоначална стойност 0**
   - **брояч за решените упражнения - с първоначална стойност 0**
   - **сумата на всички оценки - с първоначална стойност 0**
   - **коя е последната задача - с първоначална стойност празен текст**
   - **дали се е провалил или не**

```
int failedTimes = 0;
int solvedProblemsCount = 0;
double gradesSum = 0;
string lastProblem = "";
bool isFailed = true;
```

3. Създайте **while** цикъл, който продължава докато **броя на незадоволителни оценки е по-малък от числото, което сте прочели от конзолата**. При **всяко** повторение на цикъла, прочетете **името на задачата и оценката за нея**.

```
while (failedTimes < failedThreshold)
{
    string problemName = Console.ReadLine();
    if ("Enough" == problemName)
    {
        isFailed = false;
        break;
    }
}
```

4. В случай, че получите команда **Enough** променете стойността на **`isfailed`** на **`true`** и **прекратете цикъла.**
5. При **всяко повторение на цикъла**, **прибавете** оценката на Марин към **сбора на всичките му оценки** и увеличете брояча за **оценките**. Ако оценката е **по-ниска или равна на 4** увеличете брояча за **незадоволителни оценки**. Презапишете името на **последната задача**.

```
while (failedTimes < failedThreshold)
{
    string problemName = Console.ReadLine();
    if ("Enough" == problemName)
    {
        isFailed = false;
        break;
    }
    int grade = int.Parse(Console.ReadLine());
    if (grade <= 4)
    {
        failedTimes++;
    }
    gradesSum += grade;
    solvedProblemsCount++;
    lastProblem = problemName;
}
```

6. След цикъла ако броя **незадоволителни оценки** е достигнал **максималните незадоволителни оценки**, принтирайте нужното съобщение:

```
if (isFailed)
{
    Console.WriteLine($"You need a break, {failedThreshold} poor grades.");
}
else
{
    Console.WriteLine($"Average score: {gradesSum / solvedProblemsCount:F2}");
    Console.WriteLine($"Number of problems: {solvedProblemsCount}");
    Console.WriteLine($"Last problem: {lastProblem}");
}
```

36. Почивка

Джеси е решила да събира пари за екскурзия и иска от вас да ѝ помогнете да разбере **дали ще успее да събере необходимата сума**. **Тя спестява** или **харчи част от парите** си **всеки ден**. Ако иска да **похарчи повече от наличните си пари**, то тя ще похарчи **колкото има и ще ѝ останат 0 лева**.

Вход

От конзолата се четат:

- **Пари нужни за екскурзията** - **реално число в интервала [1.00.. .25000.00]**
- **Налични пари** - **реално число в интервала [0.00... 25000.00]**

След това многократно се четат по два реда:

- **Вид действие – текст с възможности "spend" и "save".**
  - **Сумата, която ще спести/похарчи - реално число в интервала [0.01… 25000.00]**

Изход

Програмата трябва да приключи при следните случаи:

- Ако **5 последователни дни** Джеси **само харчи, на конзолата да се изпише:**
  - **"You can't save the money."**
  - **"{Общ брой изминали дни}"**
- **Ако Джеси събере парите за почивката** на **конзолата се изписва:**
  - **"You saved the money for {общ брой изминали дни} days."**

Примерен вход и изход

Вход	Изход	Обяснения	
2000 1000 spend 1200 save 2000	You saved the money for 2 days.	Пари нужни за почивката: 2000 Наличните пари: 1000 spend – изваждаме от парите следващото число 1200 - ние разполагаме с 1000, но се опитваме да похарчим 1200, тъй като не разполагаме с толкова, харчим наличните си 1000 и оставаме 0 лева. save – прибавяме към парите следващото число 2000 – разполагаме с 0, добавяме 2000 и събираме парите успешно за 2 дни.	
110 60 spend 10 spend 10 spend 10 spend 10 spend 10	You can't save the money. 5	250 150 spend 50 spend 50 save 100 save 100	You saved the money for 4 days.

Насоки

1. Прочетете входните данни от конзолата:

```
double neededMoney = double.Parse(Console.ReadLine());
double ownedMoney = double.Parse(Console.ReadLine());
```

2. Направете **две помощни променливи** в началото, които да следят **броя изминали дни и броя последователни дни, в които Джеси харчи пари**. Нека и двете променливи да бъдат с първоначална **стойност нула**:

```
int daysCounter = 0;
int spendingCounter = 0;
```

Създайте **while** цикъл, който продължава, докато парите на Джеси са по-малко от парите, които са ѝ нужни за екскурзията и броячът за последователните дни е по-малък от 5. При **всяко повторение на цикъла** четете от конзолата **два реда** - първият ред е текст - **spend** или **save**, а вторият – парите, които Джеси е спестила или похарчила. Също така увеличете брояча за дни с 1:

```
while (ownedMoney < neededMoney && spendingCounter < 5)
{
    string command = Console.ReadLine();
    double money = double.Parse(Console.ReadLine());
    daysCounter++;
}
```

3. Направете проверка дали Джеси **харчи или спестява** за дадения ден:
    a. ако **спестява**, прибавете спестените пари към нейните и **нулирайте брояча за поредните дни**;
    b. ако **харчи**, извадете от нейните пари сумата която е похарчила и **увеличете брояча за поредните дни,** в които харчи. Проверете дали парите на Джеси са станали **по-малко от нула** и ако е така, то тя е останала без пари и има нула лева.
4. След цикъла проверете дали Джеси е харчила пари в **пет последователни дни** и принтирайте съобщението. Също така проверете дали Джеси е **събрала парите** и, ако е успяла, принтирайте съответното съобщение:

```
if (spendingCounter == 5)
{
    Console.WriteLine("You can't save the money.");
    Console.WriteLine(daysCounter);
}
if (ownedMoney >= neededMoney)
{
    Console.WriteLine($"You saved the money for {daysCounter} days.");
}
```

37. Стъпки

Габи иска да започне здравословен начин на живот и си е поставила за цел да върви **10 000 стъпки всеки ден**. Някои дни обаче е много уморена от работа и ще иска да се прибере преди да постигне целта си. Напишете програма, която **чете от конзолата по колко стъпки изминава** тя всеки път като излиза през деня и **когато постигне целта си да се изписва** "**`Goal reached! Good job!`**" и колко стъпки повече е извървяла **`"{разликата между стъпките} steps over the goal!"`**

Ако иска да се **прибере преди това**, тя ще въведе **командата** "**`Going home`**" и **ще въведе стъпките, които е извървяла докато се прибира**. След което, ако не е успяла да постигне целта си, на конзолата трябва да се изпише: **`"{разликата между стъпките} more steps to reach goal."`**

Примерен вход и изход

Вход	Изход	Вход	Изход
1000 1500 2000 6500	Goal reached! Good job! 1000 steps over the goal!	1500 300 2500 3000 Going home 200	2500 more steps to reach goal.
Вход	**Изход**	**Вход**	**Изход**

1500 3000 250 1548 2000 Going home 2000	Goal reached! Good job! 298 steps over the goal!	125 250 4000 30 2678 4682	Goal reached! Good job! 1765 steps over the goal!

38. Монети

Производителите на вендинг машини искали да направят машините си да връщат възможно **най-малко монети ресто**. Напишете програма, която приема **сума** - **рестото**, което трябва да се върне и изчислява **с колко най-малко монети може да стане това**.

Примерен вход и изход

Вход	Изход	Обяснения
1.23	4	Рестото ни е 1 лев и 23 стотинки. Машината ни го връща с 4 монети: монета от 1 лев, монета от 20 стотинки, монета от 2 стотинки и монета от 1 стотинка.
2	1	Рестото ни е 2 лева. Машината ни го връща с 1 монета от 2 лева.
0.56	3	Рестото ни е 56 стотинки. Машината ни го връща с 3 монети: монета от 50 стотинки, монета от 5 стотинки и монета от 1 стотинка.
2.73	5	Рестото ни е 2 лева и 73 стотинки. Машината ни го връща с 5 монети: монета от 2 лева, монета от 50 стотинки, монета от 20 стотинки, монета от 2 стотинки и монета от 1 стотинка.

39. * Торта

Поканени сте на 30-ти рожден ден, на който рожденикът черпи с огромна торта. Той обаче не знае **колко парчета могат да си вземат гостите от нея**. Вашата задача е да напишете програма, която изчислява **броя на парчетата**, които гостите са взели, преди тя да свърши. Ще получите **размерите на тортата** (широчина и дължина – **цели числа** в интервала [1...1000]) и след това на всеки ред, до получаване на командата "**STOP**" или **докато не свърши тортата**, броят на парчетата, които гостите вземат от нея.

Бележка: Едно парче торта е с размер 1x1 см.

Да се **отпечата** на конзолата **един** от следните редове:

- **"{брой парчета} pieces are left."** - ако стигнете до **STOP** и не са свършили парчетата торта
- **"No more cake left! You need {брой недостигащи парчета} pieces more."**

Примерен вход и изход

Вход	Изход	Обяснения
10 10 20 20	No more cake left! You need 1 pieces more.	Размер на тортата: 10*10 = 100. Въвеждат се многократно брой парчета които са взети: 20+20+20+20+21=101

20 20 21		Не ни достига едно парче: 101-100=1
10 2 2 4 6 STOP	8 pieces are left.	

40. Пирамида от числа

Напишете програма, която чете цяло число **n**, въведено от потребителя, и отпечатва **пирамида от числа** като в примерите:

Вход	Изход
7	1 2 3 4 5 6 7

Вход	Изход
10	1 2 3 4 5 6 7 8 9 10

Вход	Изход
12	1 2 3 4 5 6 7 8 9 10 11 12

Вход	Изход
15	1 2 3 4 5 6 7 8 9 10 11 12 13 14 15

Насоки

1. **Прочетете** едно **цяло число от конзолата**:

```
int n = int.Parse(Console.ReadLine());
```

2. **Направете два вложени `for` цикъла,** с които да печатате пирамидата от числа, като външният цикъл ще определя **колко реда** да се отпечатат, а вътрешният – **колко числа** се принтират на съответния **ред**:

```
for (int rows = 1; rows <= n; rows++)
{
    for (int cols = 1; cols <= rows; cols++)
    {
    }
}
```

3. В отделен **брояч** пазете колко числа сте отпечатали **до момента** (и кое е текущото число). Когато стигнете **n**, излезте от двата вложени цикъла с **`break`. За да излезем и от двата цикъла трябва да използваме оператора `break` и в двата. За целта ще направим булева променлива, която да проверява дали сме излезнали от вътрешния. Отидете в началото на програмата и инициализирайте следните две променливи:**

```
int n = int.Parse(Console.ReadLine());

int current = 1;
bool isBigger = false;
```

4. Във **вътрешния for цикъл** направете проверка **дали променливата current е станала по-голяма от n**. Ако е**, променете стойността на булевата** променлива и **излезте от вътрешния цикъл**:

```
for (int rows = 1; rows <= n; rows++)
{
    for (int cols = 1; cols <= rows; cols++)
    {
        if (current > n)
        {
            isBigger = true;
            break;
        }
    }
}
```

5. **След проверката**, принтирайте променливата **`current`** в желания формат и я **увеличете с 1**. Ако сте излезли от цикъла няма да се стигне до принтиране!

```
for (int rows = 1; rows <= n; rows++)
{
    for (int cols = 1; cols <= rows; cols++)
    {
        if (current > n)
        {
            isBigger = true;
            break;
        }
        Console.Write(current + " ");
        current++;
    }
}
```

6. **В тялото на външния цикъл**, направете **проверка** дали трябва да **излезем и от него**. След това отпечатайте **един празен ред**, за да може **следващите числа да са на нов ред**. Ако сме излезли от външния цикъл няма да се стигне до изпълнение на командата **`Console.WriteLine()`**! Програмата ви трябва да изглежда по следния начин:

```
for (int rows = 1; rows <= n; rows++)
{
    for (int cols = 1; cols <= rows; cols++)
    {
        if (current > n)
        {
            isBigger = true;
            break;
        }
        Console.Write(current + " ");
        current++;
    }
    if (isBigger)
    {
        break;
    }
    Console.WriteLine();
}
```

41. Еднакви суми на четни и нечетни позиции

Напишете програма, която чете от конзолата **две шестцифрени цели числа** в диапазона от 100000 до 300000. Винаги **първото** въведено число ще бъде **по малко от второто**. На конзолата да се отпечатат на **1 ред разделени с интервал** всички числа, които се намират **между двете**, прочетени от конзолата числа и отговарят на следното **условие**:

- **сумата** от цифрите на **четни** и **нечетни** позиции да са **равни**. Ако няма числа, отговарящи на условието на конзолата не се извежда резултат.

Примерен вход и изход

Вход	Изход	Обяснения
100000 100050	100001 100012 100023 100034 100045	Първото число, което генерираме е числото 100000. Сумата от цифрите на четни позиции (жълто) е 0+0+0=0. Сумата от цифрите на нечетни позиции (зелено) е 0+0+1=1. Тъй като двете суми са различни числото не се отпечатва. Следващото, число е 100001. Сумата на четни позиции е 1+0+0=1, а на нечетни 0+0+1=1. Двете суми са равни и числото се отпечатва. Следващото число за проверка е 100002. То не отговаря на условието и не се отпечатва. …… При числото 100045 сумата от четните позиции е 5+0+0=5, а на нечетни 4+0+1=5. Двете суми са равни числото се отпечатва. И т.н.

Вход	Изход	Вход	Изход	Вход	Изход
123456 124000	123464 123475 123486 123497 123530 123541 123552 123563 123574 123585 123596 123640 123651 123662 123673 123684 123695 123750 123761 123772 123783 123794 123860 123871 123882 123893 123970 123981 123992	299900 300000	299970 299981 299992	100115 100120	*Няма изход*

Насоки

1. Прочетете входните данни от потребителя:

```
int firstNum = int.Parse(Console.ReadLine());
int secondNum = int.Parse(Console.ReadLine());
```

2. За да преминете през всички числа от интервала, направете **for** цикъл. След като сте прочели входните числа, задайте първото число за начална стойност на контролната променлива**.** Итерирайте до достигане на **второто число** като **увеличавате** стойността на контролната променлива с **1**:

```
int firstNum = int.Parse(Console.ReadLine());
int secondNum = int.Parse(Console.ReadLine());

for (int i = firstNum; i <= secondNum; i++)
{

}
```

3. Вземете числото на текущата позиция **като текст, като използвате метода** `.ToString()`:

```
for (int i = firstNum; i <= secondNum; i++)
{
    string currentNum = i.ToString();
}
```

4. За да обходите всяка цифра от числото, направете **for** цикъл. След като е взето като текст, вземете неговата дължина с `.Length`. Итерирайте до достигане на **дължината на числото** като **увеличавате** стойността на контролната променлива с **1**:

```
string currentNum = i.ToString();
for (int j = 0; j < currentNum.Length; j++)
{
}
```

5. Продължете към дописване на логиката за намиране на сумата на четна и нечетна позиция за всяко число. Декларирайте по една променлива за четната и нечетна сума. За да получите точната числова стойност на цифрите използвайте метода **`int.Parse()`**.

```
for (int i = firstNum; i <= secondNum; i++)
{
    string currentNum = i.ToString();
    int oddSum = 0;
    int evenSum = 0;
    for (int j = 0; j < currentNum.Length; j++)
    {
        int currentDigit = int.Parse(currentNum.ToString());
    }
}
```

6. За да намерите цифрите, които се намират на четна позиция, използвайте условна **if** конструкция, проверете дали индекса му е четно число като го разделите модулно на 2(**index % 2**), ако е четно, добавете го към сумата на четните, ако не е, към сумата на нечетните.

```
for (int i = firstNum; i <= secondNum; i++)
{
    string currentNum = i.ToString();
    int oddSum = 0;
    int evenSum = 0;
    for (int j = 0; j < currentNum.Length; j++)
    {
        int currentDigit = int.Parse(currentNum[j].ToString());
        if (j % 2 == 0)
        {
            evenSum += currentDigit;
        }
        else
        {
            oddSum += currentDigit;
        }
    }
}
```

7. След като сте намерили сумата на цифрите на четни и нечетни позиции, проверете дали са равни, ако са, принтирайте числото. Програмата ви трябва да изглежда по следния начин:

```
int firstNum = int.Parse(Console.ReadLine());
int secondNum = int.Parse(Console.ReadLine());

for (int i = firstNum; i <= secondNum; i++)
{
    string currentNum = i.ToString();
    int oddSum = 0;
    int evenSum = 0;
    for (int j = 0; j < currentNum.Length; j++)
    {
        int currentDigit = int.Parse(currentNum[j].ToString());
        if (j % 2 == 0)
        {
            evenSum += currentDigit;
        }
        else
        {
            oddSum += currentDigit;
        }
    }
    if (oddSum == evenSum)
    {
        Console.Write(i + " ");
    }
}
```

42. Суми прости и непрости числа

Напишете програма, която чете от конзолата цели числа в диапазона от **-2,147,483,648** до **2,147,483,647**, докато не се получи команда "**stop**". Да се намери **сумата** на всички въведени **прости** и сумата на всички въведени **непрости** числа. Тъй като по дефиниция от математиката отрицателните числа не могат да бъдат прости, ако на входа се подаде **отрицателно** число да се изведе следното съобщение "**`Number is negative.`**". В този случай въведено число се игнорира и не се прибавя към нито една от двете суми, а програмата продължава своето изпълнение, очаквайки въвеждане на следващо число.

На **изхода** да се отпечатат на два реда **двете намерени суми** в следния формат:

`"Sum of all prime numbers is:` {prime numbers sum}"

`"Sum of all non prime numbers is:` {nonprime numbers sum}"

Примерен вход и изход

Вход	Изход	Обяснения
3 9 0 7 19 4 stop	Sum of all prime numbers is: 29 Sum of all non prime numbers is: 13	Първото въведено число е 3. То е просто и го прибавяме съм сумата на простите числа. Следващото число е 9. То не е просто и го прибавяме към сумата на непростите числа. Числото 0 не е просто число и го прибавяме към сумата на непростите числа. Сумата става 9+0=9. Следващите две числа са 7 и 19. Те са прости и всяко едно от тях го прибавяме към сумата на простите числа. 3+7=10 и 10+19=29. Следва числото 4, което не е просто и го прибавяме към съответната сума 9+4=13. Получаваме команда stop. Програмата прекъсва своето изпълнение и отпечатваме двете суми.

Вход	Изход
30 83 33 -1 20 stop	Number is negative. Sum of all prime numbers is: 83 Sum of all non prime numbers is: 83

Вход	Изход
0 -9 0 stop	Number is negative. Sum of all prime numbers is: 0 Sum of all non prime numbers is: 0

43. Train the Trainers

Курсът "Train the trainers" е към края си и финалното оценяване наближава. Вашата задача е да помогнете на журито което ще оценява презентациите, като напишете програма в която да изчислява **средната оценка** от представянето на **всяка една презентация** от даден студент, а накрая **средният успех от всички тях**.

От конзолата на първият ред се прочита броят на хората в журито **n** - **цяло число в интервала [1…20]**

След това на отделен ред се прочита името на презентацията - **текст**

За всяка една презентация на нов ред се четат **n - на брой оценки - реално число в интервала [2.00…6.00]**

След изчисляване на средната оценка за конкретна презентация, на конзолата се печата

`"{името на презентацията} - {средна оценка}."`

След получаване на команда **"`Finish`"** на конзолата се печата **"`Student's final assessment is {среден успех от всички презентации}.`"** и програмата приключва.

Всички оценки трябва да бъдат форматирани до **втория знак** след десетичната запетая.

Примерен вход и изход

Вход	Изход	Обяснения

2 While-Loop 6.00 5.50 For-Loop 5.84 5.66 Finish	While-Loop - 5.75. For-Loop - 5.75. Student's final assessment is 5.75.	2 – броят на хората в журито следователно ще получаваме по 2 оценки на презентация. (6.00 + 5.50) / 2 = 5.75 (5.84 + 5.66) / 2 = 5.75 (6.00 + 5.50 + 5.84 + 5.66) / 4 = 5.75	
Вход	**Изход**	**Вход**	**Изход**
3 Arrays 4.53 5.23 5.00 Lists 5.83 6.00 5.42 Finish	Arrays - 4.92. Lists - 5.75. Student's final assessment is 5.34.	2 Objects and Classes 5.77 4.23 Dictionaries 4.62 5.02 RegEx 2.88 3.42 Finish	Objects and Classes - 5.00. Dictionaries - 4.82. RegEx - 3.15. Student's final assessment is 4.32.

44. Генератор за пароли

Да се напише програма, която чете две цели числа ***n*** и ***l***, въведени от потребителя, и генерира по азбучен ред всички възможни **пароли**, които се състоят от следните **5 символа**:

- Символ 1: цифра от **1** до ***n***.
- Символ 2: цифра от **1** до ***n***.
- Символ 3: малка буква измежду първите ***l*** букви на латинската азбука.
- Символ 4: малка буква измежду първите ***l*** букви на латинската азбука.
- Символ 5: цифра от 1 до **n**, по-голяма от първите 2 цифри.

Вход

Входът се чете от конзолата и се състои от две **цели числа *n*** и ***l*** в интервала [**1**...**9**], по едно на ред.

Изход

На конзолата трябва да се отпечатат **всички пароли** по **азбучен ред**, разделени с **интервал**.

Примерен вход и изход

Вход	Изход
2 4	11aa2 11ab2 11ac2 11ad2 11ba2 11bb2 11bc2 11bd2 11ca2 11cb2 11cc2 11cd2 11da2 11db2 11dc2 11dd2
3 1	11aa2 11aa3 12aa3 21aa3 22aa3
3 2	11aa2 11aa3 11ab2 11ab3 11ba2 11ba3 11bb2 11bb3 12aa3 12ab3 12ba3 12bb3 21aa3 21ab3 21ba3 21bb3 22aa3 22ab3 22ba3 22bb3
4 2	11aa2 11aa3 11aa4 11ab2 11ab3 11ab4 11ba2 11ba3 11ba4 11bb2 11bb3 11bb4 12aa3 12aa4 12ab3 12ab4 12ba3 12ba4 12bb3 12bb4 13aa4 13ab4 13ba4 13bb4 21aa3 21aa4 21ab3 21ab4 21ba3 21ba4 21bb3 21bb4 22aa3 22aa4 22ab3 22ab4 22ba3 22ba4 22bb3

	22bb4 23aa4 23ab4 23ba4 23bb4 31aa4 31ab4 31ba4 31bb4 32aa4 32ab4 32ba4 32bb4 33aa4 33ab4 33ba4 33bb4

45. Специални числа

Да се напише програма, която **чете едно цяло число N**, въведено от потребителя, и генерира всички възможни **"специални" числа** от **1111** до **9999**. За да бъде **"специално"** едно число, то трябва да отговаря на **следното условие**:

- **N да се дели на всяка една от неговите цифри без остатък.**

Пример: при **N = 16**, **2418** е специално число:

- **16 / 2** = 8 **без остатък**
- **16 / 4** = 4 **без остатък**
- **16 / 1** = 16 **без остатък**
- **16 / 8** = 2 **без остатък**

Вход

Входът се чете от конзолата и се състои от **едно цяло число** в интервала [**1**…**600000**]

Изход

На конзолата трябва да се отпечатат **всички "специални" числа**, разделени с **интервал**

Примерен вход и изход

Вход	Изход	Коментари
3	1111 1113 1131 1133 1311 1313 1331 1333 3111 3113 3131 3133 3311 3313 3331 3333	3 / 1 = 3 без остатък 3 / 3 = 1 без остатък 3 / 3 = 1 без остатък 3 / 3 = 1 без остатък
11	1111	
16	1111 1112 1114 1118 1121 1122 1124 1128 1141 1142 1144 1148 1181 1182 1184 1188 1211 1212 1214 1218 1221 1222 1224 1228 1241 1242 1244 1248 1281 1282 1284 1288 1411 1412 1414 1418 1421 1422 1424 1428 1441 1442 1444 1448 1481 1482 1484 1488 1811 1812 1814 1818 1821 1822 1824 1828 1841 1842 1844 1848 1881 1882 1884 1888 2111 2112 2114 2118 2121 2122 2124 2128 2141 2142 2144 2148 2181 2182 2184 2188 2211 2212 2214 2218 2221 2222 2224 2228 2241 2242 2244 2248 2281 2282 2284 2288 2411 2412 2414 2418 2421 2422 2424 2428 2441 2442 2444 2448 2481 2482 2484 2488 2811 2812 2814 2818 2821 2822 2824 2828 2841 2842 2844 2848 2881 2882 2884 2888 4111 4112 4114 4118 4121 4122 4124 4128 4141 4142 4144 4148 4181 4182 4184 4188 4211 4212 4214 4218 4221 4222 4224 4228 4241 4242 4244 4248 4281 4282 4284 4288 4411 4412 4414 4418 4421 4422 4424 4428 4441 4442 4444 4448 4481 4482 4484 4488 4811 4812 4814 4818 4821 4822 4824 4828 4841 4842 4844 4848 4881 4882 4884 4888 8111 8112 8114 8118 8121 8122 8124 8128 8141 8142 8144 8148 8181 8182 8184 8188 8211 8212 8214 8218 8221 8222 8224 8228 8241 8242 8244 8248 8281 8282 8284 8288 8411 8412 8414 8418 8421 8422 8424 8428 8441 8442 8444 8448 8481 8482 8484 8488 8811 8812 8814 8818 8821 8822 8824 8828 8841 8842 8844 8848 8881 8882 8884 8888	