# Упражнения: Повторения (цикли)

Задачи за упражнение в клас и за домашно към курса [„Основи на програмирането“ @ СофтУни](https://softuni.bg/courses/programming-basics).

## Празно Visual Studio решение (Blank Solution)

1. Създайте празно решение (**Blank Solution**) във Visual Studio за да организирате кода от задачите за упражнение. Целта на този **blank solution** e да съдържа **по един проект за всяка задача** от упражненията.



1. Задайте **да се стартира по подразбиране текущия проект** (не първият в решението). Кликнете с десен бутон на мишката върху **Solution 'Loops'** 🡪 **[Set StartUp Projects…]** 🡪 **[Current selection]**.

## Числа от 1 до 100

Напишете програма, която отпечатва числата от 1 до 100, по едно на ред.

|  |  |
| --- | --- |
| **вход** | **изход** |
| (няма) | 1  2  3  …  98  99  100 |

**Подсказки**:

1. Създайте **нов проект** в съществуващото Visual Studio решение – конзолна C# програма. Задайте подходящо име на проекта, например “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:



1. **Стартирайте** програмата с [Ctrl+F5] и я **тествайте**:



1. **Тествайте** решението си в **judge системата**: <https://judge.softuni.bg/Contests/Practice/Index/154#0>. Трябва да получите **100 точки** (напълно коректно решение).

## 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 |

**Тествайте** решението си в **judge системата**: <https://judge.softuni.bg/Contests/Practice/Index/154#1>.

**Подсказка**: можете да завъртите for-цикъл от 1 до 1000 и да проверите всяко число дали завършва на 7. Едно число num завършва на 7, когато (num % 10 == 7).

## Всички латински букви

Напишете програма, която отпечатва всички букви от латинската азбука: a, b, c, …, z.

**Тествайте** решението си в **judge системата**: <https://judge.softuni.bg/Contests/Practice/Index/154#2>.

**Подсказка**: можете да завъртите for-цикъл от 'a' до 'z' (освен числа може да въртите в цикъл и букви).

## 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. Примери:

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **вход** | **изход** |  | **вход** | **изход** |  | **вход** | **изход** |  | **вход** | **изход** |  | **вход** | **изход** |
| 2  10  20 | 30 | 3  -10  -20  -30 | -60 | 4  45  -20  7  11 | 43 | 1  999 | 999 | 0 | 0 |

**Тествайте** решението си в **judge системата**: <https://judge.softuni.bg/Contests/Practice/Index/154#3>.

**Подсказки**:

* Първо въведете едно число n (броят числа, които предстои да бъдат въведени).
* Инициализирайте sum = 0 (в началото няма още прочетени числа, и съответно сумата е празна).
* В цикъл n **пъти** прочетете по едно цяло число num и го прибавете към сумата (sum = sum + num).
* Накрая в sum трябва да се е запазила сумата на прочетените числа. Отпечатайте я.

## Най-голямо число

Напишете програма, която чете n-на брой **цели числа** (n > 0), въведени от потребителя, и намира **най-голямото** измежду тях. Първо се въвежда броят числа n, а след това самите n числа, по едно на ред. Примери:

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **вход** | **изход** |  | **вход** | **изход** |  | **вход** | **изход** |  | **вход** | **изход** |  | **вход** | **изход** |
| 2  100  99 | 100 | 3  -10  20  -30 | 20 | 4  45  -20  7  99 | 99 | 1  999 | 999 | 2  -1  -2 | -1 |

**Тествайте** решението си в **judge системата**: <https://judge.softuni.bg/Contests/Practice/Index/154#4>.

**Подсказки**:

* Първо въведете едно число n (броят числа, които предстои да бъдат въведени).
* Въведете от конзолата първото число. Сложете текущият максимум max да е прочетеното число.
* В цикъл n**-1 пъти** прочетете по едно цяло число num. Ако прочетеното число num е по-голямо от текущият максимум max, запомнете num в max.
* Накрая в max трябва да се е запазило най-голямото число. Отпечатайте го.

## Най-малко число

Напишете програма, която чете n-на брой **цели числа** (n > 0), въведени от потребителя, и намира **най-малкото** измежду тях. Първо се въвежда броят числа n, а след това самите n числа, по едно на ред. Примери:

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **вход** | **изход** |  | **вход** | **изход** |  | **вход** | **изход** |  | **вход** | **изход** |  | **вход** | **изход** |
| 2  100  99 | 99 | 3  -10  20  -30 | -30 | 4  45  -20  7  99 | -20 | 1  999 | 999 | 2  -1  -2 | -2 |

**Тествайте** решението си в **judge системата**: <https://judge.softuni.bg/Contests/Practice/Index/154#5>.

**Подсказки**: задачата е абсолютно аналогична с предходната.

## 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" + **сумата**; иначе печата "No" + **разликата**. Разликата се изчислява като положително число (по абсолютна стойност). Примери:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **вход** | **изход** | **коментар** |  | **вход** | **изход** | **коментар** |
| 2  10  90  60  40 | Yes, sum = 100 | 10+90 = 60+40 = 100 | 2  90  9  50  50 | No, diff = 1 | 90+9 ≠ 50+50  Difference = |99-100| = 1 |

**Тествайте** решението си в **judge системата**: <https://judge.softuni.bg/Contests/Practice/Index/154#6>.

**Подсказки**:

* Въведете n.
* Въведете първите n числа (**лявата** половина) и ги сумирайте.
* Въведете още n числа (**дясната** половина) и ги сумирайте.
* Изчислете **разликата** между сумите по абсолютна стойност: Math.Abs(leftSum - rightSum).
* Ако разликата е 0, отпечатайте "Yes" + **сумата**; иначе отпечатайте "No" + **разликата**.

## 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 отпечата "Yes" + **сумата**; иначе да се отпечата "No" + **разликата**. Разликата се изчислява по абсолютна стойност. Примери:

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **вход** | **изход** | **коментар** |  | **вход** | **изход** | **коментар** |  | **вход** | **изход** | **коментар** |
| 4  10  50  60  20 | Yes  Sum = 70 | 10+60 = 50+20 = 70 | 4  3  5  1  -2 | No  Diff = 1 | 3+1 ≠ 5-2  Diff = |4-3| = 1 | 3  5  8  1 | No  Diff = 2 | 5+1 ≠ 8  Diff = |6-8| = 2 |

**Тествайте** решението си в **judge системата**: <https://judge.softuni.bg/Contests/Practice/Index/154#7>.

**Подсказки**: Въведете числата едно по едно и изчислете двете **суми** (числа на **четни** позиции и числа на **нечетни** позиции). Както в предходната задача, изчислете абсолютна стойност на разликата и отпечатайте резултата ("Yes" + **сумата** при разлика 0 или "No" + **разликата** в противен случай).

## 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 |

Примери:

|  |  |  |
| --- | --- | --- |
| **вход** | **изход** | **коментар** |
| hello | 6 | e + o = 2 + 4 = 6 |
| hi | 3 | i = 3 |
| bamboo | 9 | a + o + o = 1 + 4 + 4 = 9 |
| beer | 4 | e + e = 2 + 2 = 4 |

**Тествайте** решението си в **judge системата**: <https://judge.softuni.bg/Contests/Practice/Index/154#8>.

**Подсказки**:

* Прочетете входния текст s. Нулирайте сумата.
* Завъртете цикъл от 0 до s.Length-1 (дължината на текста -1).
* Проверете всяка буква s[i] дали е гласна и съответно добавете към сумата стойността й.

## \*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).

Примери:

|  |  |  |
| --- | --- | --- |
| **вход** | **изход** | **коментари** |
| 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 |  |

**Тествайте** решението си в **judge системата**: <https://judge.softuni.bg/Contests/Practice/Index/154#9>.

**Подсказка**: изчислете **сумата** на всички елементи и **най-големият** от тях и проверете търсеното условие.

## \*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**”}

Примери:

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **вход** | **изход** |  | **вход** | **изход** |  | **вход** | **изход** |  | **вход** | **изход** |
| 6  **2**  3  **5**  4  **2**  1 | OddSum=9, OddMin=2, OddMax=5, EvenSum=8, EvenMin=1, EvenMax=4 | 2  **1.5**  -2.5 | OddSum=1.5, OddMin=1.5, OddMax=1.5, EvenSum=-2.5, EvenMin=-2.5, EvenMax=-2.5 | 1  **1** | OddSum=1, OddMin=1, OddMax=1, EvenSum=0, EvenMin=No, EvenMax=No | 0 | OddSum=0, OddMin=No, OddMax=No, EvenSum=0, EvenMin=No, EvenMax=No |
| **вход** | **изход** |  | **вход** | **изход** |  | **вход** | **изход** |  | **вход** | **изход** |
| 5  **3**  -2  **8**  11  **-3** | OddSum=8, OddMin=-3, OddMax=8, EvenSum=9, EvenMin=-2, EvenMax=11 |  | 4  1.5  **1.75**  1.5  **1.75** | OddSum=3, OddMin=1.5, OddMax=1.5, EvenSum=3.5, EvenMin=1.75, EvenMax=1.75 |  | 1  **-5** | OddSum=-5, OddMin=-5, OddMax=-5, EvenSum=0, EvenMin=No, EvenMax=No |  | 3  **-1**  -2  **-3** | OddSum=-4, OddMin=-3, OddMax=-1, EvenSum=-2, EvenMin=-2, EvenMax=-2 |

**Тествайте** решението си в **judge системата**: <https://judge.softuni.bg/Contests/Practice/Index/154#10>.

**Подсказки**:

* 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.
* Работете с **дробни числа** (не цели). Сумата, минимумът и максимумът също са дробни числа.
* Използвайте **неутрална начална стойност** при намиране на минимум / максимум, например **1000000000.0** и **-1000000000.0**. Ако получите накрая неутралната стойност, печатайте “**No**”.

## Еднакви двойки

Дадени са 2\*n-на брой числа. Първото и второто формират **двойка**, третото и четвъртото също и т.н. Всяка двойка има **стойност** – сумата от съставящите я числа. Напишете програма, която проверява **дали всички двойки имат еднаква стойност** или печата **максималната разлика** между две последователни двойки. Ако всички двойки имат еднаква стойност, отпечатайте "**Yes, value={Value}**" + **стойността**. В противен случай отпечатайте "**No, maxdiff={Difference}**" + **максималната разлика**. Примери:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **вход** | **изход** | **коментари** |  | **вход** | **изход** | **коментари** |
| 3  1  2  0  3  4  -1 | Yes, value=3 | стойности = {3, 3, 3}  еднакви стойности | 2  1  2  2  2 | No, maxdiff=1 | стойности = {3, 4}  разлики = {1}  макс. разлика = 1 |
| 4  1  1  3  1  2  2  0  0 | No, maxdiff=4 | стойности = {2, 4, 4, 0}  разлики = {2, 0, 4}  макс. разлика = 4 | 1  5  5 | Yes, value=10 | стойности = {10}  една стойност  еднакви стойности |
| 2  -1  0  0  -1 | Yes, value=-1 | стойности = {-1, -1}  еднакви стойности |  | 2  -1  2  0  -1 | No, maxdiff=2 | стойности = {1, -1}  разлики = {2}  макс. разлика = 2 |

**Тествайте** решението си в **judge системата**: <https://judge.softuni.bg/Contests/Practice/Index/154#11>.

**Подсказки**:

* Прочитайте входните числа **по двойки**. За всяка двойка пресмятайте **сумата**.
* Докато четете входните двойки, за всяка двойка без първата пресмятайте **разликата с предходната**. За целта пазете в отделна променлива сумата на предходната двойка.
* Намерете **най-голямата разлика** между две двойки. Ако е **0**, печатайте “**Yes**” иначе “**No**” + разликата.

# Упражнения: Графични и Web приложения

## \* Чертане с костенурка – графично GUI приложение

Целта на следващото упражнение е да си поиграем с една **библиотека за рисуване**, известна като “**графика с костенурка**” (**turtle graphics**). Ще изградим **графично приложение**, в което ще **рисуваме различни фигури**, придвижвайки нашата “**костенурка**” по екрана чрез операции от типа “отиди напред 100 позиции”, “завърти се надясно на 30 градуса”, “отиди напред още 50 позиции”. Приложението ще изглежда приблизително така:



1. Запознайте се с **концепцията за рисуване** “**Turtle Graphics**”. Можете да разгледате следните източници:

* Дефиниция на понятието “turtle graphics” – <http://c2.com/cgi/wiki?TurtleGraphics>
* Статия за “turtle graphics” в Wikipedia – <https://en.wikipedia.org/wiki/Turtle_graphics>
* Интерактивен онлайн инструмент за чертане с костенурка – <https://blockly-games.appspot.com/turtle>

1. Създайте нов **C# Windows Forms проект**:



1. Инсталирайте **NuGet** пакета "Nakov.TurtleGraphics" към вашият Windows Forms проект.

От Visual Studio може да се добавят **външни библиотеки** (пакети) към вашите C# проекти. Те добавят допълнителна функционалност към вашите приложения. Официалното хранилище (repository) за C# библиотеки се поддържа от Microsoft и се нарича **NuGet** ([www.nuget.org](http://www.nuget.org)).

Кликнете върху проекта в **Solution Explorer** и изберете **[Manage NuGet Packages…]**:



Ще се отвори прозорец за търсене и инсталиране на **NuGet** пакети. Потърсете пакети по ключова дума “nakov”. Ще излязат няколко пакета. Изберете пакет “Nakov.TurtleGraphics”. Натиснете **[Install]** за да го инсталирате към вашия C# проект:



Към вашият C# проект вече в включена външната библиотека “Nakov.TurtleGraphics”. Тя дефинира един клас Turtle, с който представлява **костенурка за рисуване**. За да го използвате, трябва да добавите в C# кода за вашата форма (Form1.cs) следния код най-отгоре в началото на файла:

|  |
| --- |
| using Nakov.TurtleGraphics; |

1. Сега сложете **три бутона** във формата и нагласете **имената** и **свойствата** им както е посочено по-долу:



1. Кликнете два пъти върху бутона **[Draw]**, за да напишете кода, който да се изпълни при натискането му. Напишете следния код:

|  |
| --- |
| private void buttonDraw\_Click(object sender, EventArgs e)  {  Turtle.Rotate(30);  Turtle.Forward(200);  Turtle.Rotate(120);  Turtle.Forward(200);  Turtle.Rotate(120);  Turtle.Forward(200);  } |

Този код мести и върти костенурката, която в началото е в центъра на екрана (в средата на формата), и чертае равностранен триъгълник. Може да го редактирате и да си поиграете с него.

1. **Стартирайте** приложението с **[Ctrl+F5]**. Тествайте го дали работи (натиснете **[Draw]** бутона няколко пъти):



1. Сега можете да напишете **по-сложна програма за костенурката**:

|  |
| --- |
| // Assign a delay to visualize the drawing process  Turtle.Delay = 200;  // Draw a equilateral triangle  Turtle.Rotate(30);  Turtle.Forward(200);  Turtle.Rotate(120);  Turtle.Forward(200);  Turtle.Rotate(120);  Turtle.Forward(200);  // Draw a line in the triangle  Turtle.Rotate(-30);  Turtle.PenUp();  Turtle.Backward(50);  Turtle.PenDown();  Turtle.Backward(100);  Turtle.PenUp();  Turtle.Forward(150);  Turtle.PenDown();  Turtle.Rotate(30); |

1. **Стартирайте** приложението с **[Ctrl+F5]**. Тествайте дали работи новата програма за костенурката:



Вече костенурката чертае по-сложни фигури чрез приятно анимирано движение.

1. Напишете кода и за останалите два бутона. Целта на бутона **[Reset]** е да изтрие графиката и да започне да чертае на чисто:

|  |
| --- |
| private void buttonReset\_Click(object sender, EventArgs e)  {  Turtle.Reset();  } |

1. Целта на бутона **[Show / Hide Turtle]** е да показва или скрива костенурката:

|  |
| --- |
| private void buttonShowHideTurtle\_Click(object sender, EventArgs e)  {  if (Turtle.ShowTurtle)  {  Turtle.ShowTurtle = false;  this.buttonShowHideTurtle.Text = "Show Turtle";  }  else  {  Turtle.ShowTurtle = true;  this.buttonShowHideTurtle.Text = "Hide Turtle";  }  } |

1. **Стартирайте** приложението с **[Ctrl+F5]**. Тествайте дали работят правилно всички бутони.

## \* Чертане на шестоъгълник с костенурката

Добавете бутон **[Hexagon]**, който чертае правилен шестоъгълник:



**Подсказка**: В цикъл повторете 6 пъти следното: ротация на 60 градуса; движение напред 100.

## \* Чертане на звезда с костенурката

Добавете бутон **[Star]**, който чертае звезда с 5 върха (**петолъчка**) като на фигурата по-долу:



**Подсказка**:

* Сменете цвета: Turtle.PenColor = Color.Green.
* В цикъл повторете 5 пъти следното: движение напред 200, ротация на 144 градуса.

## \* Чертане на спирала с костенурката

Добавете бутон **[Spiral]**, който чертае спирала с 20 върха като на фигурата по-долу:



**Подсказка**: Чертайте в цикъл като движите напред и завъртате. С всяка стъпка увеличавайте постепенно дължината на движението напред и завъртайте на 60 градуса.

## \* Чертане на слънце с костенурката

Добавете бутон **[Sun]**, който чертае слънце с 36 върха като на фигурата по-долу:



## \* Чертане на спирален триъгълник с костенурката

Добавете бутон **[Triangle]**, който три триъгълника с по 22 върха като на фигурата по-долу:



**Подсказка**: Чертайте в цикъл като движите напред и завъртате. С всяка стъпка увеличавайте с 10 дължината на движението напред и завъртайте на 120 градуса. Повторете 3 пъти за трите триъгълника.

# Изпитни задачи от минали издания на курса

## \* Дядо Ставри

*Четвърта задача от изпита на 17 септември 2017. Тествайте решението си* [***тук***](https://judge.softuni.bg/Contests/Practice/Index/784#3)

Дядо Ставри има казан за ракия и всеки ден вари от любимата си напитка. Ако вари **N дена** и всеки ден получава **различно количество ракия** и **алкохолен градус**, да се намери завсичките дни **общото количество ракия и градуса на получената смес.**

### Вход

От конзолата се четат **поредица от числа, всяко на отделен ред**:

* На **първия ред** – N – брой дни – **цяло число в интервала** **[1...300]**
* За **всеки един ден** на отделен ред :
* **количество на ракията** – **реално число в интервала [1.00...500.00]**
* **градус на получената напитка - реално число в интервала [10.00...99.00]**

### Изход

Да се отпечатат на конзолата **3 реда**, както следва:

* **Първи ред** – да се отпечатва съобщение форматирано до втори знак „**Liter**:{общо литрите}“
* **Втори ред** – да се отпечатва съобщение форматирано до втори знак „**Degrees**:{градусите на общата смес} ;
* **Трети ред** – да се отпечатва:
* Ако градусът е по-малко от 38, да се отпечата съобщение „Не е добре, трябва преваряване!“. 🡪"**Not good, you should baking**!".
* Ако е от 38 до 42, да се отпечата съобщение „Супер!“ 🡪"**Super**!"
* Ако е повече от 42, да се отпечата „Разреждане с дестилирана вода!“🡪 "**Dilution with distilled water**!"

### Примерен вход и изход

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Вход** | **Изход** | **Обяснения** | **Вход** | **Изход** |
| 3  100  45  50  55  150  36 | Liter: 300.00  Degrees: 42.17  Dilution with distilled water! | **Първи ден:** общото=100 литра  100 \* 45=4500  **Втори ден:** общото= 100+50=150;  4500+55\*50=7250;  **Трети ден:** общото= 150+150=300;  7250+150\*36=12650;  **градусите**=12650 / 300=42.166666… | 2  200  43  200  40 | Liter: 400.00  Degrees: 41.50  Super! |

## \* Торта

*Четвърта задача от изпита на 03 септември 2017. Тествайте решението си* [***тук***](https://judge.softuni.bg/Contests/Practice/Index/750#3)

Поканен сте на **30-тия рожден ден на Кендрик Ламар**, но като част от организацията. Той иска да покани много гости и планира да вземе **огромна торта**. Иска да знае обаче колко **парчета** могат да се вземат от нея. Всеки път когато някой си вземе, трябва да се **отчита**, че **едно** или **повече** парчета са взети. Така трябва да се продължи **докато не свърши тортата** или **докато гостите не спрат да си взимат**.

Напишете **програма**, която да изчислява броя **останали** парчета. Ще получите размерите на тортата и след това на всеки ред, докато не получите командата **STOP** или докато не свърши тортата (**pieces < 0**), ще получавате цяло число, равно на взетите парчета. Накрая трябва да се отпечата **колко парчета са останали** или, **че са свършили**.

**Note: Тортата е с правоъгълна форма. Едно парче е с размер 1х1.**

### Вход

Потребителят въвежда следните данни на отделни редове:

1. **Широчина на тортата – цяло число в интервала [1...1000];**
2. **Дължина на тортата – цяло число в интервала [1...1000];**
3. На следващите редове **(докато не получите командата STOP)** – **брой парчета, които се взимат от тортата – цели числа в интервала [1...1000];**

### Изход

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

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

### Примерен вход и изход

|  |  |
| --- | --- |
| **Вход** | **Изход** |
| 10  10  20  20  20  20  21 | No more cake left! You need 1 pieces more. |
| **Вход** | **Изход** |
| 10  2  2  4  6  STOP | 8 pieces are left. |

## \* Разход на енергия

*Четвърта задача от изпита на 23 юли 2017. Тествайте решението си* [***тук***](https://judge.softuni.bg/Contests/Practice/Index/715#3)

След финалите всички танцьори се връщат към регулярните си тренировки в залата. **Всички тренират определен брой дни по определени часове.** **Да се изчисли колко средно енергия е останала неизхабена за всички тренировъчни дни** като се има предвид че:

|  |  |
| --- | --- |
|  | **Проценти изхабена енергия на човек** |
| **Четна тренировка и четни часове** | **68%** |
| **Нечетна тренировка и четни часове** | **49%** |
| **Четна тренировка и нечетни часове** | **65%** |
| **Нечетна тренировка и нечетни часове** | **30%** |

**Всеки танцьор започва със 100% енергия.**

### Вход

От конзолата се четат **поредица от числа**, всяко **на отделен ред**:

1. **N – брой тренировъчни дни – цяло число в интервала [0 … 31].**
2. **А – брой танцьори – цяло число в интервала [0 … 100].**

**За всеки тренировъчен ден** се прочита **на отделен ред**:

* **Часовете, които са тренирали** – **цяло число в интервала [0 … 12].**

### Изход

Отпечатва се **1 ред** в следния формат:

**АКО** средния разход на енергия за тренировъчните дни е **50% или по - малко**

* **“They feel good! Energy left:** {**останала енергия**}**”**.

**AКО** средния разход на енергия за тренировъчните дни е **над 50%**

* **“They are wasted! Energy left:** {**останала енергия**}**”**.

**Резултатът трябва да се форматира до втория знак след десетичната запетая**.

|  |  |  |
| --- | --- | --- |
| **Вход** | **Изход** | **Обяснения** |
| 4  10  5  8  2  3 | They are wasted! Energy left: 47.00 | **Нечетен ден и нечетни часове** – 10 \* 30 = 300 общо отдадена енергия от всички.  **Четен ден и четни часове** – 68 \* 10 = 680    **Нечетен ден и четни часове** = 49 \* 10 = 490  **Четен ден и нечетни часове** – 65 \* 10 = 650  **Обща енергия** за всички дни на танцьорите =  100 \* 10 \* 4 = 4000  Неизхабена енергия общо за **всички** дни -  4000 – (300 + 680 + 490 + 650) = 1880    Неизхабена енергия на **танцьор** **за ден** –  1880 / 10 / 4 = 47 |
| **Вход** | **Изход** | **Обяснения** |
| 4  10  1  1  1  1 | They feel good! Energy left: 52.50 | **Нечетен ден и нечетни часове – 10\*30 = 300**    **Четен ден и нечетни часове – 65\*10 =650**  **Нечетен ден и нечетни часове – 10\*30 = 300**  **Четен ден и нечетни часове – 65\*10 = 650**  **Обща енергия** за всички дни на танцьорите –  100 \* 10 \* 4 = 4000  Неизхабена енергия общо за **всички** дни -  4000 – (650 + 650 + 300 + 300) = 2100  Неизхабена енергия на **танцьор** **за ден** –  2100 / 10 / 4 = 52.5 |

## \* Парична награда

*Четвърта задача от изпита на 25 юни 2017. Тествайте решението си* [***тук***](https://judge.softuni.bg/Contests/Practice/Index/642#3)

Сашо, Пешо, Гошо, Иво и Стоян печелят състезание за най-добре изготвен проект и получават парична награда. **Проектът се разделя на части. Различните части на проекта, носят различен брой точки. За всяка четна част (2, 4, 6…) има бонус – точките умножени по две.** Да се напише програма,която пресмята **каква сума са получили участниците. От конзолата се прочита на колко части е разделен проектът, паричната награда за всяка точка и колко точки се дават за всяка част.**

### Вход

От конзолата се четат **поредица от числа**, всяко **на отделен ред**:

1. **На колко части е разделен проектът –** **цяло число** **в интервала [1 … 50].**
2. **Паричната награда за една точка** – **реално число в интервала [0.00 … 10000.00].**

**За всяка част от проекта** се прочита **на отделен ред**:

* **Точките, които се получават** – **цяло число в интервала [0 … 100].**

### Изход

Отпечатва се **1 ред** в следния формат:

* **“The project prize was {сумата} lv.”**.

**Резултатът трябва да се форматира до втория знак след десетичната запетая**.

### Примерен вход и изход

|  |  |  |
| --- | --- | --- |
| **Вход** | **Изход** | **Обяснения** |
| 7  100.5  2  4  5  1  1  7  3 | The project prize was 3517.50 lv. | Проектът е разделен на **7 части.**  **Точки**: 2 + (4 \* 2) + 5 + (1 \* 2) + 1 + (7 \* 2) + 3 = **35**  **Награда:** 100.5 \* 35 = **3517.50 лв.** |

## \* Футболен турнир

*Четвърта задача от изпита на 07 май 2017. Тествайте решението си* [***тук***](https://judge.softuni.bg/Contests/Practice/Index/540#3)

Екипът на СофтУни си организира футболен турнир. Първоначално прочитаме от **конзолата капацитета на стадиона и броят на всички фенове**. След това **за всеки фен се чете секторът, в който се намира**. Феновете на **първия отбор** са в **сектор А и Б**, а на **втория – В и Г**. Да се напише програма, която **изчислява процентите на феновете във всеки сектор, спрямо общия брой фенове на стадиона, както и общият процент на феновете за двата отбора, спрямо капацитета на стадиона. Общият брой на феновете НЕ надвишава капацитета на стадиона.**

### Вход

От конзолата се четат **поредица от числа**, всяко **на отделен ред**:

1. **Капацитетът на стадиона** – **цяло число в интервала [1 … 10000]**;
2. **Броят на всички фенове** – **цяло число в интервала [1 … 10000]**.

**За всеки един фен**, **на отделен ред** се прочита:

1. **Секторът, на който се намира** – **текст** - **“A”, “B”, “V” и “G”**.

### Изход

Да се отпечатат на конзолата **5 реда**, всеки от които съдържа **процент между 0.00% и 100.00%**, **форматирани до втората цифра след десетичната запетая**:

1. **Процентът на феновете в сектор А**;
2. **Процентът на феновете в сектор Б**;
3. **Процентът на феновете в сектор В**;
4. **Процентът на феновете в сектор Г**;
5. **Процентът на всички фенове, спрямо капацитета на стадиона**.

### Примерен вход и изход:

|  |  |  |  |
| --- | --- | --- | --- |
| **Вход** | **Изход** | **Обяснения** | |
| 76  10  A  V  V  V  G  B  A  V  B  B | 20.00%  30.00%  40.00%  10.00%  13.16% | Феновете в сектор А са **2**. Изчисляваме какъв процент са от **общия брой фенове – 10** на стадиона.  2 / 10 \* 100 = **20.00%**  Феновете в сектор B са **3**. По същия начин изчисляваме **техния процент** спрямо общия брой на феновете.  3 / 10 \* 100 = **30.00%**  Феновете в сектор V са **4**. По същия начин изчисляваме **техния процент** спрямо общия брой на феновете.  4 / 10 \* 100 = **40.00%**  В сектор G има само **1** фен. По същия начин изчисляваме **какъв процент е спрямо общия брой на феновете.**  1/10 \* 100 = **10.00%**  Изчисляваме процентът на всички фенове, спрямо капацитетът на стадиона:  10/ 76 \* 100 = **13.16%** | |
| **Вход** | **Изход** | **Вход** | **Изход** |
| 93  16  A  V  G  G  B  B  G  B  A  B  B  B  A  B  B  A | 25.00%  50.00%  6.25%  18.75%  17.20% | 1000  12  A  A  V  V  A  G  A  A  V  G  V  A | 50.00%  0.00%  33.33%  16.67%  1.20% |