# Упражнения: Повторения (цикли)

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

## Числа от 1 до 100

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

### Примерен вход и изход:

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

**Подсказки**:

1. Създайте **нов проект** в същщото Visual Studio решение – конзолна Node.js програма. Задайте подходящо име на проекта, например “numbers-1-to-100”.
2. Отидете в тялото на функцията numbers1To100() и напишете решението на задачата.

function numbers1To100() {

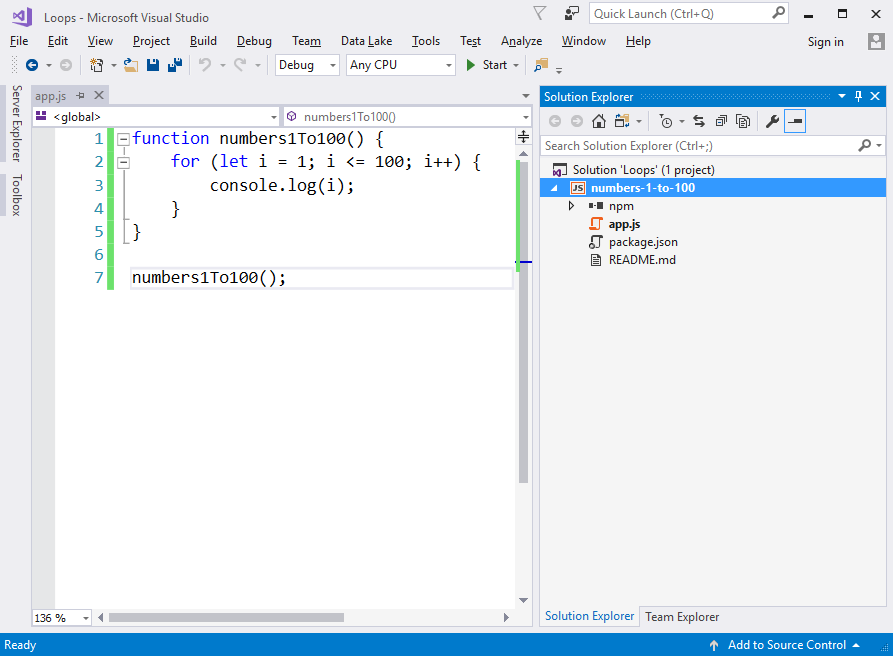
//Your code goes here ...

}

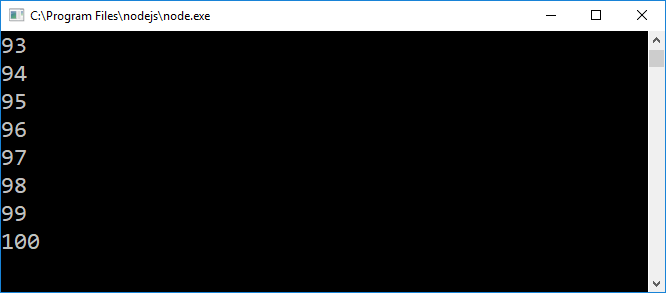
//Call the function to execute the code inside

numbers1To100();

Можете да си помогнете с кода от картинката по-долу:



1. **Стартирайте** програмата с [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**.

### Примерен вход и изход:

|  |  |
| --- | --- |
| **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**: Проверете кодовете на буквите в [ASCII таблицата](http://www.asciitable.com/) и използвайте метода String.fromCharCode().

## Сумиране на числа

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

* Първият аргумент(ред) – n, представлява броят на числата.
* Следващите n-на брой аргумента(реда) са цели числа.

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

### Примерен вход и изход:

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **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, започвайки от 1,** 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.

Използвайте следния код:

function sumNumbers(args) {

//Get the first element (n) with args[0]

//Iterate trough the elements using args[i]

}

sumNumbers(["3", "-10", "-20", "-30"]);

## 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.

### Примерен вход и изход:

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **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).
* Сложете текущият максимум max да е Number.NEGATIVE\_INFINITY.
* В цикъл до n(включително) **пъти** прочетете по едно цяло число num. Ако прочетеното число num е по-голямо от текущия 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 като аргументи, и намира **най-малкото** измежду тях. Първо се въвежда броят числа n, а след това самите n числа.

### Примерен вход и изход:

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **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, използвайте Number.POSITIVE\_INFINITY.

## Лява и дясна сума

Да се напише програма, която чете **2\***n-на брой цели числа, подадени от потребителя като аргументи, 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** |  | **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 (включително) числа (**лявата** половина) и ги сумирайте.
* Въведете още n числа (**дясната** половина, от n+1 до 2\*n) 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 като аргументи, и проверява дали **сумата от числата на четни позиции** е равна на **сумата на числата на нечетни позиции**. При равенство да се отпечата "Yes", "Sum = " + **сумата**; иначе да се отпечата "No", "Diff = " + **разликата**. Разликата се изчислява по абсолютна стойност.

### Примерен вход и изход:

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **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 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 |

### 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 |
| 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 (дължината на текста).
* 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).

### Примерен вход и изход:

|  |  |  |
| --- | --- | --- |
| **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-големия** 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**”}

### Примерен вход и изход:

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **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.
* Сумата, минимумът и максимумът са дробни числа.
* Използвайте Number.POSITIVE\_INFINITY и Number.NEGATIVE\_INFINITY при намиране на минимум / максимум. Ако получите накрая стойност infinity, 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**).

### Примерен вход и изход:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **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.

# Изпитни задачи от минали издания на курса

## Хистограма

*Четвърта задача от междинния изпит на 6 март 2016. Тествайте решението си* [***тук***](https://judge.softuni.bg/Contests/Practice/Index/169#3)*.*

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(аргумент) от входа стои цялото число n (1 ≤ n ≤ 1000) – брой числа. На следващите n **реда(аргумента)** 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** |  | **Вход** | **Изход** |  | **Вход** | **Изход** |  | **Вход** | **Изход** |  | **Вход** | **Изход** |
| **3**  1  2  999 | 66.67%  0.00%  0.00%  0.00%  33.33% | **4**  53  7  56  999 | 75.00%  0.00%  0.00%  0.00%  25.00% | **7**  800  801  250  199  399  599  799 | 14.29%  28.57%  14.29%  14.29%  28.57% | **9**  367  99  200  799  999  333  555  111  9 | 33.33%  33.33%  11.11%  11.11%  11.11% | **14**  53  7  56  180  450  920  12  7  150  250  680  2  600  200 | 57.14%  14.29%  7.14%  14.29%  7.14% |

## Деление без остатък

*Четвърта задача от изпита на 26 март 2016. Тествайте решението си* [***тук***](https://judge.softuni.bg/Contests/Practice/Index/179#3)*.*

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 = 1**0** 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(аргумент) 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** |  | **Вход** | **Изход** |
| **10**  680  2  600  200  800  799  199  46  128  65 | 70.00%  10.00%  50.00% | **3**  3  6  9 | 33.33%  100.00%  0.00% |

## Умната Лили

*Четвърта задача от изпита на 24 април 2016. Тествайте решението си* [***тук***](https://judge.softuni.bg/Contests/Practice/Index/181#3)

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(аргумента)**, въведени от потребителя, на отделни редове:

* **Възрастта** на Лили – **цяло число** в интервала **[1...77]**
* **Цената на пералнята** – число в интервала **[1.00...10 000.00]**
* **Единична цена** **на играчка** – **цяло** **число** в интервала **[0...40]**

### 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! {М}“** – 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** | **Коментари** |
| 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** |