

Лаб: Повторения с цикли – While-цикъл

Задачи за упражнение и домашно към курса "[Основи на програмирането](#)" в СофтУни.

Тествайте решението си в judge системата: <https://judge.softuni.bg/Contests/Index/2407#0>

1. Четене на думи

Напишете функция, която чете елементите на масив, 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

2. Парола

Напишете функция, 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.
- при въвеждане на правилна парола: отпечатваме "Welcome {username}!".

Примерен вход и изход

ВХОД	ИЗХОД
(["Nakov", "1234", "Pass", "1324", "1234"])	Welcome Nakov!

ВХОД	ИЗХОД
(["Gosho", "secret", "secret"])	Welcome Gosho!

Насоки

- Инициализирайте две променливи **username** и **password**, които ще съдържат потребителското име и паролата:

```
let username = input[0];  
let password = input[1];
```

- Инициализирайте променлива **data**, която ще държи въведената от потребителя парола за вход:

```
let data= input[2];
```

3. Инициализирайте променлива **counter**, която ще държи индекса на текущия елемент в масива. Тъй като вече сме присвоили първите три елемента, ще сложим **counter** да е равно на 3.

```
let index = 3;
```

4. В **while** цикъл, до въвеждане на валидна парола, четете нова и повишавайте **counter** с 1:

```
while (data !== password) {  
    data = input[index];  
    index++;  
}
```

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

```
function password(input) {  
    let username = input[0];  
    let password = input[1];  
    let data = input[2];  
    let index = 3;  
    while (data !== password) {  
        data = input[index];  
        index++;  
    }  
    console.log(`Welcome ${username}!`);  
}
```

3. Сума от числа

Напишете функция, която чете цяло число от масив 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
------	-------

вход	изход
------	-------

(["100", "10", "20", "30", "40"])	100
---	-----

(["20", "1", "2", "3", "4", "5", "6"])	21
--	----

4. Редица числа $2k + 1$

Напишете програма, която чете число n , въведено от потребителя и отпечатва **всички числа** $\leq n$ 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. Създайте променлива, която ще е брояч и има **първоначална стойност 1**.
2. Създайте **while** цикъл, който се повтаря докато **брояча е по-малък** от числото, което сте прочели от конзолата.
3. При всяко повторение на цикъла **принтирайте стойността на брояча** и му **прибавяйте дадената стойност**.

5. Баланс по сметка

Напишете функция, която пресмята колко общо пари има в сметката, след като направите определен брой вноски. Във всеки елемент 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.

Всички суми, които се печатат, трябва да се форматира 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: 100.00 Total: 174.93

вход	изход
(["120", "45.55", "-150"])	Increase: 120.00 Increase: 45.55 Invalid operation! Total: 165.55

6. Най-голямо число

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

Примерен вход и изход

вход	изход	вход	изход	вход	изход	вход	изход	вход	изход
(["100", "99", "80", "70", "Stop"])	100	(["-10", "20", "-30", "Stop"])	20	(["45", "-20", "7", "99", "Stop"])	99	(["999", "Stop"])	999	(["-1", "-2", "Stop"])	-1

7. Най-малко число

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

Примерен вход и изход

вход	изход	вход	изход	вход	изход	вход	изход	вход	изход
(["100", "99", "80", "70", "Stop"])	70	(["-10", "20", "-30", "Stop"])	-30	(["45", "-20", "7", "99", "Stop"])	-20	(["999", "Stop"])	999	(["-1", "-2", "Stop"])	-2

8. 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 отпечата **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	вход	изход
(["Gosho", "5", "5.5", "6", "Stop"])	Gosho graduated. Average grade: 5.53	(["Mimi", "5", "6", "5", "Stop"])	Mimi has been excluded at 8 grade

"5.43", "5.5", "6", "5.55", "5", "6", "6", "5.43", "5"])		"6", "5", "6", "6", "2", "3"])	
--	--	---	--