

Работа с вложени цикли

По-сложни задачи



СофтУни

Преподавателски екип



SoftUni



Софтуерен университет

<https://softuni.bg>

1. Преговор
2. Вложени цикли
3. Решаване на задачи





Преговор

1. Колко пъти ще се изпише "SoftUni" на конзолата след изпълнението на следния код:

```
let i = 0;  
while(i <= 5){  
  console.log("SoftUni");  
  i++;  
}
```

5

0

6

4

2. Колко пъти ще се изпише "SoftUni" на конзолата след изпълнението на следния код:

```
let i = 0;
while(i == 0){
  console.log("SoftUni");
  if(i == 1)
    break;
}
```

0

1

Безброй
много пъти

100000

3. Какъв ще е резултатът от изпълнението на следния код:

```
let i = 0;
while (i < 6){
  i++;
  if (i % 2 == 0)
    console.log(i);
}
```

024

24

246

123456

4. Какъв ще е резултатът от изпълнението на следния код:

```
let i = 0;
while (i < 4) {
  switch(i) {
    case 1:
      console.log(i);
    case 2:
      console.log(i);
      break;
    case 3:
      console.log(i);
      break;
  }
  i++;
}
```

1

11

1123

2



Вложени цикли

По-сложни комбинаторни задачи

Пример – часовник (1)

Часовете се променят
когато минутите
надвишат 59

19:03

Докато минутите се
променят часовете
остават същите



Как да си направим часовник с код?

Демо

Пример – часовник (2)

- Външният цикъл отговаря за часовете, а вътрешният за минутите

```
for (let h = 0; h <= 23; h++) {  
  for (let m = 0; m <= 59; m++) {  
    console.log(`${h}:${m}`);  
  }  
}
```

PROBLEMS	OUTPUT	DEBUG CONSOLE
	10:51	
	10:52	
	10:53	
	10:54	
	10:55	
	10:56	
	10:57	
	10:58	
	10:59	
	11:0	
	11:1	
	11:2	
	11:3	
	11:4	
	11:5	
	11:6	
	11:7	
	11:8	
	11:9	
	11:10	

- За всяка итерация на външния цикъл вложения се изпълнява **n** - на брой пъти

```
for (let i = 0; i < n; i++)  
  for (let j = 0; j < n; j++)
```

... Имената на променливите трябва да бъдат различни



Таблица за умножение – условие

- Отпечатайте на конзолата таблицата за умножение за числата от 1 до 10

- Изход:

```
PROBLEMS  OUTPUT  DEBUG CONSOLE
C:\Program Files\nodejs\node
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
2 * 1 = 2
2 * 2 = 4
2 * 3 = 6
2 * 4 = 8
2 * 5 = 10
2 * 6 = 12
2 * 7 = 14
2 * 8 = 16
2 * 9 = 18
2 * 10 = 20
3 * 1 = 3
```



Таблица за умножение – решение

```
for (let x = 1; x <= 10; x++) {  
  for (let y = 1; y <= 10; y++) {  
    let product = x * y;  
    console.log(`${x} * ${y} = ${product}`);  
  }  
}
```

- За прекъсване на вложени цикли, използваме булеви променливи.

```
let flag = false;  
for (let i = 0; i < n; i++)  
  for (let j = 0; j < n; j++)  
    if (condition)  
      flag = true;  
      break;  
    if (flag)  
      break;
```

Външният цикъл ще се прекъсне, само ако стойността на flag бъде true

- Напишете функция, която проверява всички възможни комбинации от двойка числа в даден интервал
 - Ако се намери комбинация, чийто **сбор от числата е равен** на дадено **магическо число** на изхода **се отпечатва съобщение** и програмата приключва изпълнение
 - Ако не се намери **ниито една комбинация**, отговаряща на условието се отпечатва **съобщение, че не е намерено**

Сума от две числа – условие (2)

- Примерен вход и изход:

1
10
5



Combination N:4 (1 + 4 = 5)

23
24
20



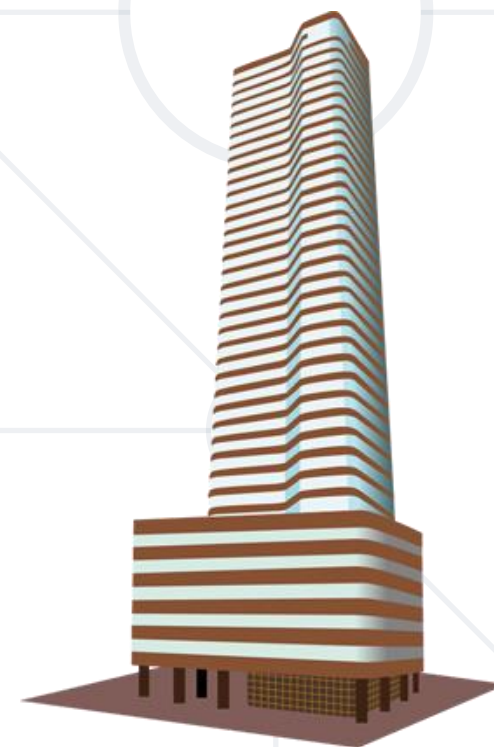
4 combinations - neither equals 20

Сума от две числа – решение

```
let startingNumber = Number(input[0]);
let finalNumber = Number(input[1]);
let magicNumber = Number(input[2]);
let combinations = 0;
let isFound = false;
for (let i = startingNumber; i <= finalNumber; i++)
  for (let j = startingNumber; j <= finalNumber; j++)
    combinations++;
    if (i + j === magicNumber)
      console.log(`Combination N:${combinations} (${i} + ${j} =
        ${magicNumber})`);
      isFound = true;
      break;
  if (isFound)
    break;
// Finish logic
```

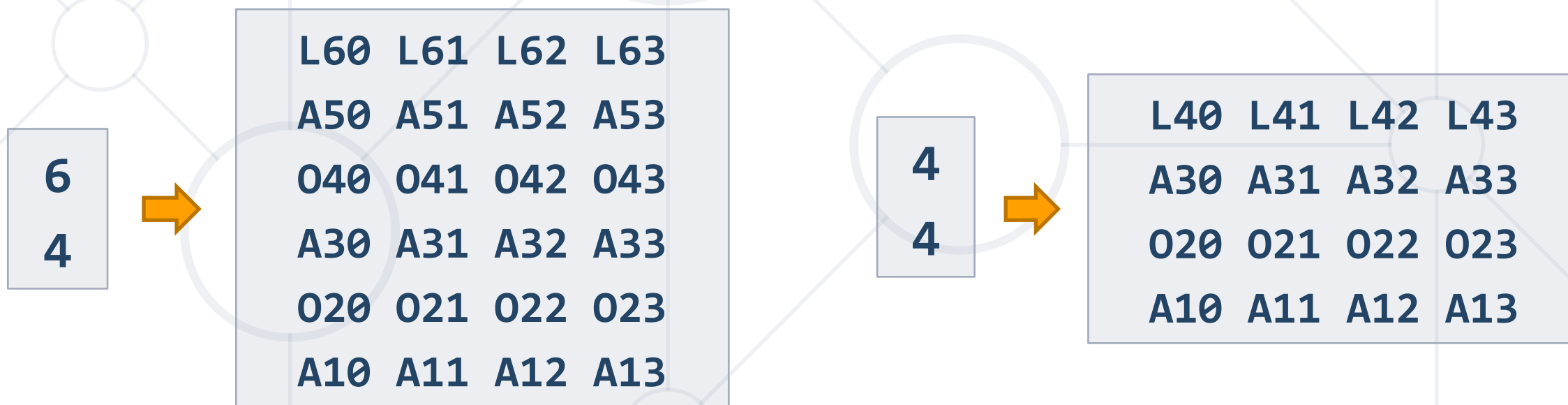
Ако намерим
комбинация, прекъсваме
вътрешният цикъл

- Напишете функция, която извежда номерата на стаите в една сграда (в низходящ ред)
 - На всеки **четен** етаж има само **офиси**
 - На всеки **нечетен** етаж има само **апартаменти**
- Етажите се означават по следния начин:
 - Апартаменти: "А{номер на етажа}{номер на апартамента}"
 - Офиси: "О{номер на етажа}{номер на офиса}"
 - Номерата им винаги започват с 0



Сграда – условие (2)

- На последният етаж винаги има големи апартаменти, които се означават с 'L', вместо с 'A'
- Ако има само един етаж, то има само големи апартаменти
- Примерен вход и изход:



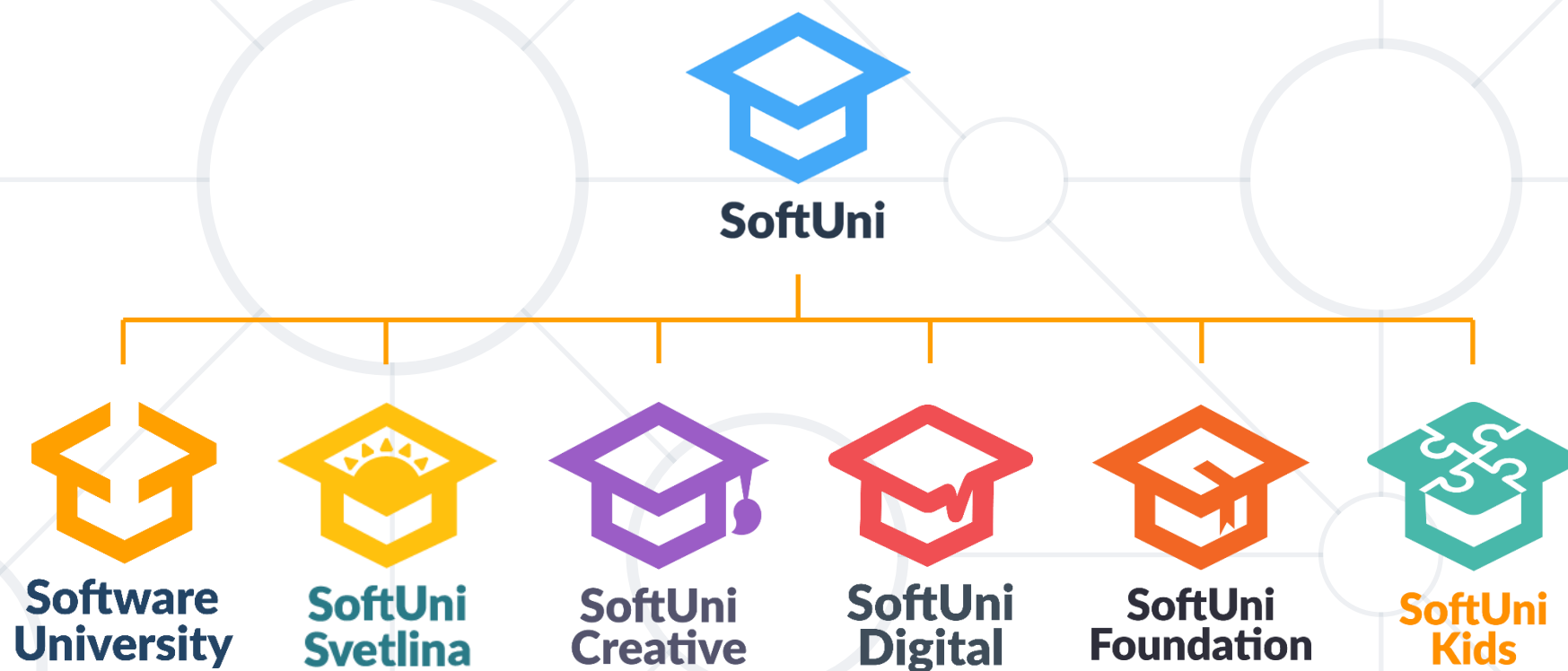
```
let floors = Number(input[0]);
let rooms = Number(input[1]);
for (let i = floors; i >= 1; i--) {
  let printLine = "";
  for (let j = 0; j < rooms; j++) {
    if (i == floors)
      printLine += `L${i}${j} `;
    // TODO: print according to floor number
  }
  console.log(printLine);
}
```

Вложеният цикъл
итерира стаите

- Какво представляват вложените цикли
- Конструкция на вложени цикли
- Прекъсване на вложени цикли



Въпроси?



- Този курс (презентации, примери, демонстрационен код, упражнения, домашни, видео и други активи) представлява **защитено авторско съдържание**
- Нерегламентирано копиране, разпространение или използване е незаконно
- © СофтУни – <https://softuni.org>
- © Софтуерен университет – <https://softuni.bg>



- Софтуерен университет – качествено образование, професия и работа за софтуерни инженери
 - softuni.bg
- Фондация "Софтуерен университет"
 - softuni.foundation
- Софтуерен университет @ Facebook
 - facebook.com/SoftwareUniversity
- Дискусионни форуми на СофтУни
 - forum.softuni.bg



Software University

