

# Работа с вложени цикли

По-сложни задачи



СофтУни

Преподавателски екип



SoftUni



Софтуерен университет

<http://softuni.bg>

# Имате въпроси?

sli.do

#pb-jan

1. Преговор
2. Вложени цикли





**Преговор**

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

```
int i = 0;  
while(i <= 5) {  
    cout << "SoftUni" << endl;  
    i++;  
}
```

5

0

4

6

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

```
int i = 0;
while(i == 0) {
    cout << "SoftUni" << endl;
    if(i == 1)
        break;
}
```

0

1

100000

Безброй  
много пъти

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

```
int i = 0;
while (i < 6) {
    i++;
    if (i % 2 == 0)
        cout << i << endl;
}
```

024

24

246

123456



# **Вложени цикли**

## **По-сложни комбинаторни задачи**



# Пример – часовник (1)

Часовете се променят  
когато минутите  
надвишат 59

Докато минутите се  
променят часовете  
остават същите

19:03



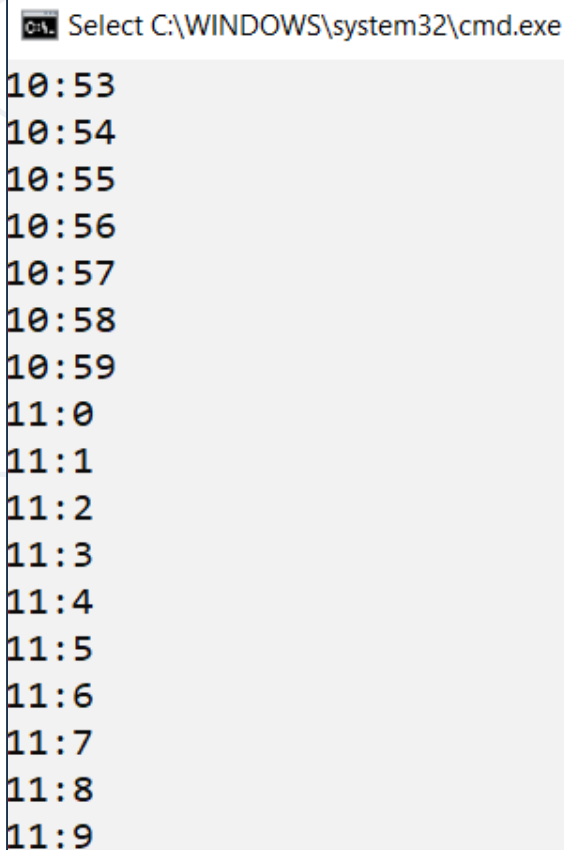
**Как може да си направим часовник с код?**

**Демо**

# Пример – часовник (2)

- Външният цикъл отговаря за часовете, а вътрешния за минутите

```
for (int h = 0; h <= 23; h++) {  
    for (int m = 0; m <= 59; m++) {  
        cout << h << ":" << m << endl;  
    }  
}
```



```
Select C:\WINDOWS\system32\cmd.exe  
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
```

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

Имената на  
итераторите трябва  
да бъдат различни

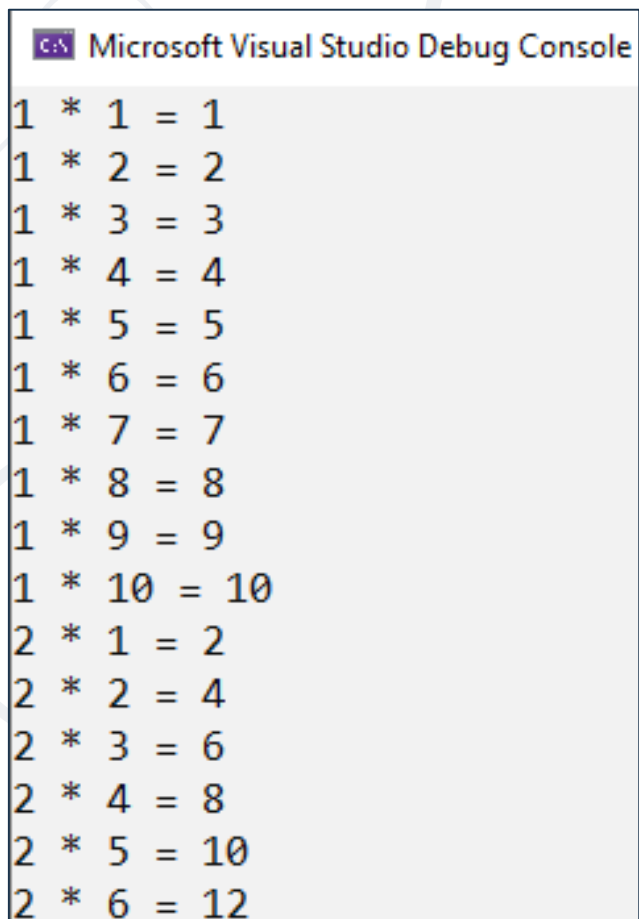
- За всяка итерация на външния цикъл вложения се изпълнява **n** - на брой пъти



# Таблица за умножение - условие

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

- Изход:



```
Microsoft Visual Studio Debug Console
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
```



# Таблица за умножение - решение

```
for (int x = 1; x <= 10; x++) {  
    for (int y = 1; y <= 10; y++) {  
        int product = x * y;  
        cout << x << " * " << y << " = " << product << endl;  
    }  
}
```

- За прекъсване на вложени цикли, използваме булеви променливи.

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

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

- Напишете програма, 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

```
int startingNumber, finalNumber, magicNumber;
cin >> startingNumber >> finalNumber >> magicNumber;
int combinations = 0;
bool isFound = false;
for (int i = startingNumber; i <= finalNumber; i++) {
    for (int j = startingNumber; j <= finalNumber; j++) {
        combinations++;
        if (i + j == magicNumber) {
            cout << "Combination N:" << combinations
                << " (" << i << " + " << j << " = "
                << magicNumber << ")" << endl;
            isFound = true;
            break;
        }
    }
    if (isFound)
        break;
} // Finish logic
```

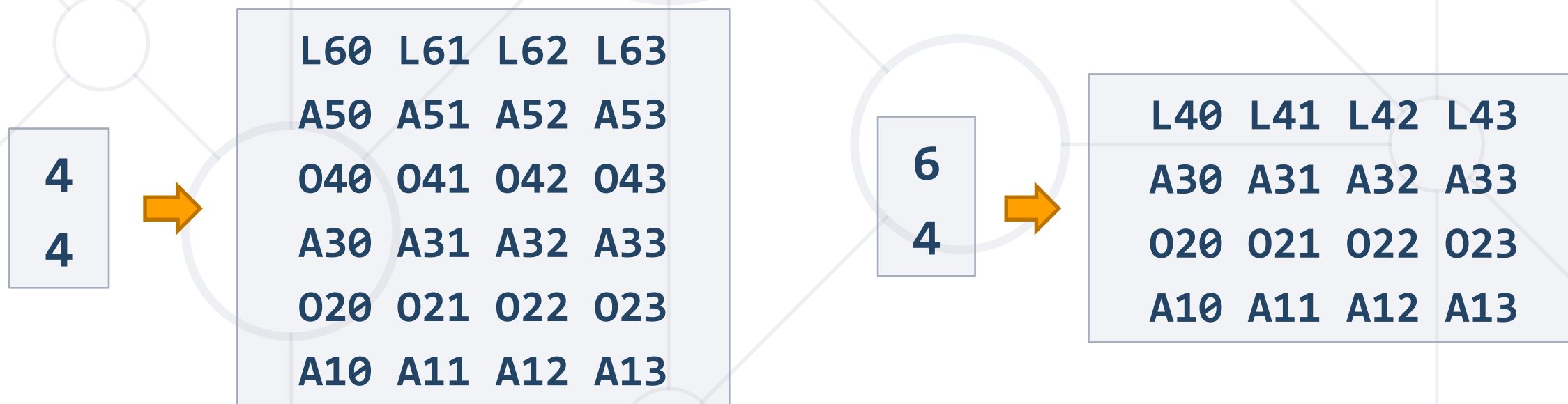
Ако намерим  
комбинация, прекъсваме  
вътрешният цикъл

- Напишете програма, 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:



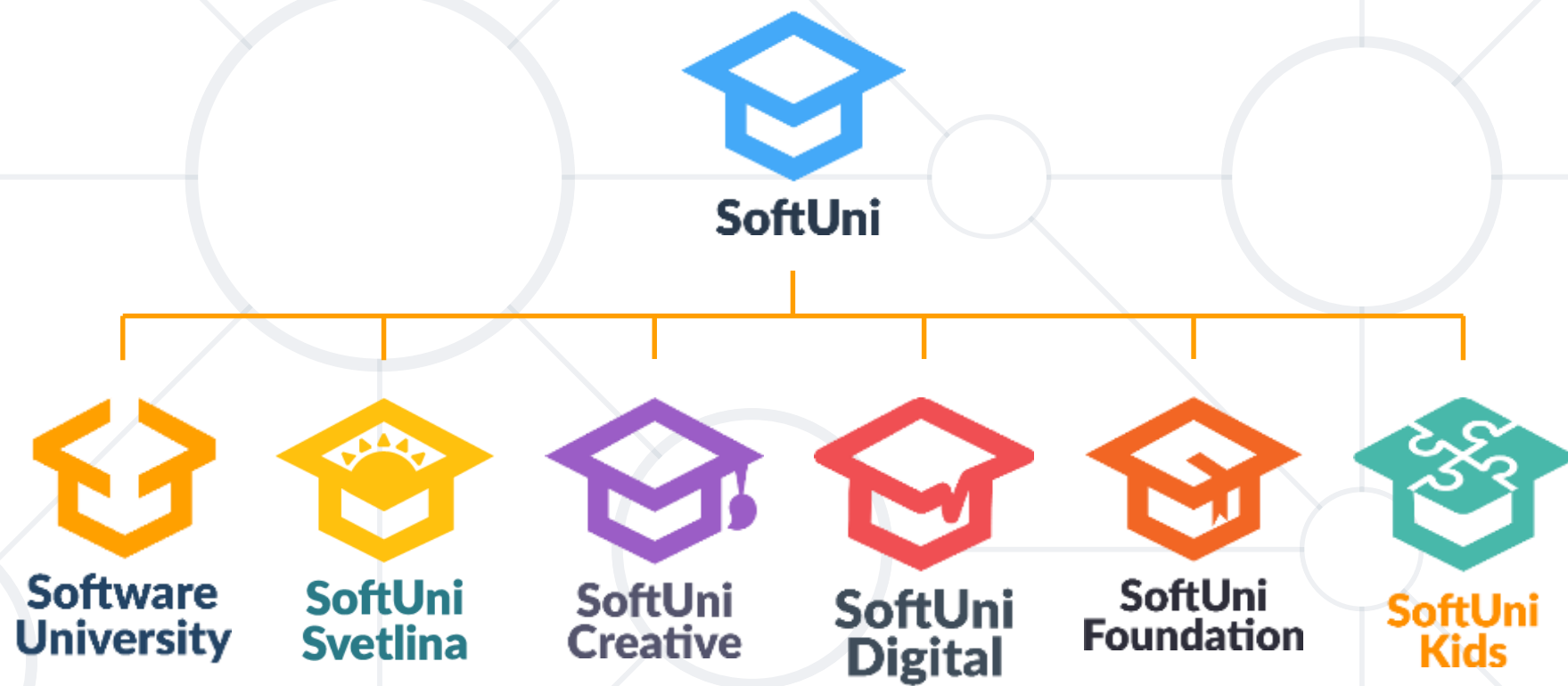
```
int floors; cin >> floors;
int rooms; cin >> rooms;
for (int i = floors; i >= 1; i--) {
    for (int j = 0; j < rooms; j++) {
        if (i == floors)
            cout << "L" << i << j << " ";
        // TODO: print according to floor number
    }
    cout << endl;
}
```

Вложеният цикъл  
итерира стаяте

- 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](http://softuni.bg)
- Фондация "Софтуерен университет"
  - [softuni.foundation](http://softuni.foundation)
- Софтуерен университет @ Facebook
  - [facebook.com/SoftwareUniversity](https://facebook.com/SoftwareUniversity)
- Дискуссионни форуми на СофтУни
  - [forum.softuni.bg](http://forum.softuni.bg)



Software University

