Семинар - група 1

Седмица 2

1. **Глобална и локална видимост на декларации и дефиниции.**

Когато използваме блокове, за да разделяме код или пък след оператори като if, else и т.н., се нуждаем и да знаем и кога кои променливи как се виждат в кода от другите функции, оператори, променливи и т.н. Когато дефинираме блок ние казваме, че всички декларации и дефиниции са локални за него, а тези извън него са глобални за всички от неговото ниво.

```
int a = 5;
{
    int b = 6;
}
std::cout << a + b;
```

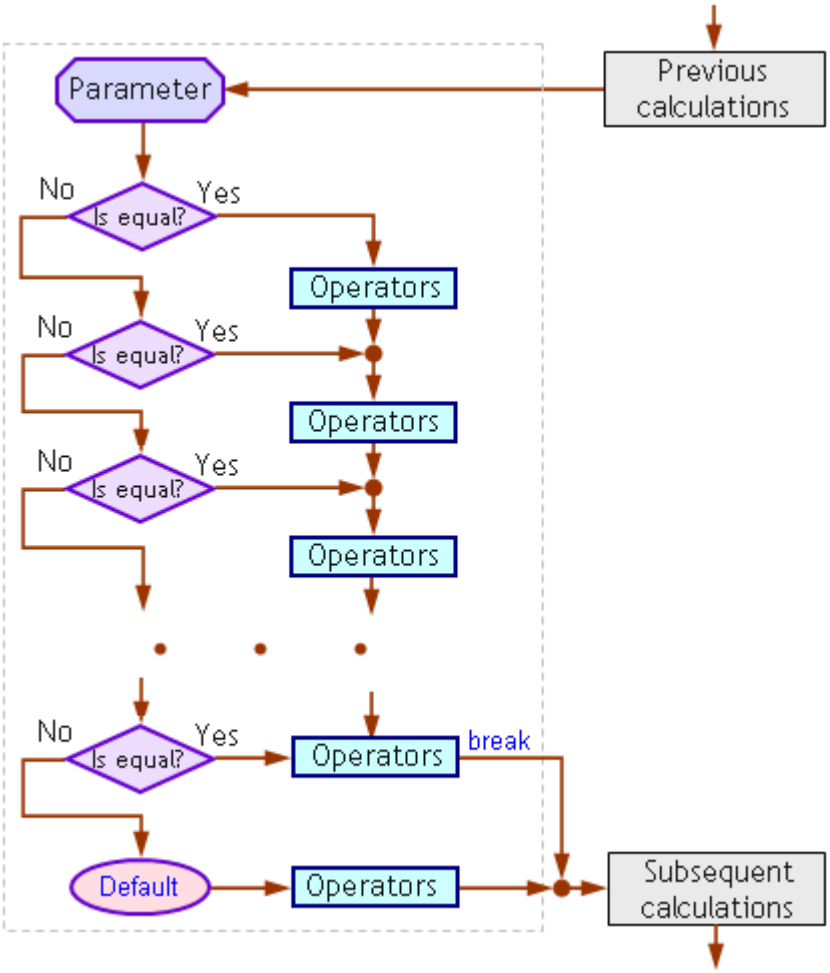
```
example.cpp: In function ‘int main()’:
example.cpp:9:22: error: ‘b’ was not declared in this scope
    9 |     std::cout << a + b;
      |                      ^
```

Когато се опитаме да компилираме съответния код, получаваме **КОМПИЛАЦИОННА** грешка, която ни казва, че променливата b не е декларирана в съответния блок /scope/. По същия начин работи и когато декларираме променливи в if-else, цикли и т.н.

2. **Оператор Switch. Оператор break.**

По подобие на оператор If и тук имаме различни действия при определено условие. Разликата обаче е, че при switch имаме сравняване на променлива с

константни стойности, които се наричат случаи /cases/.



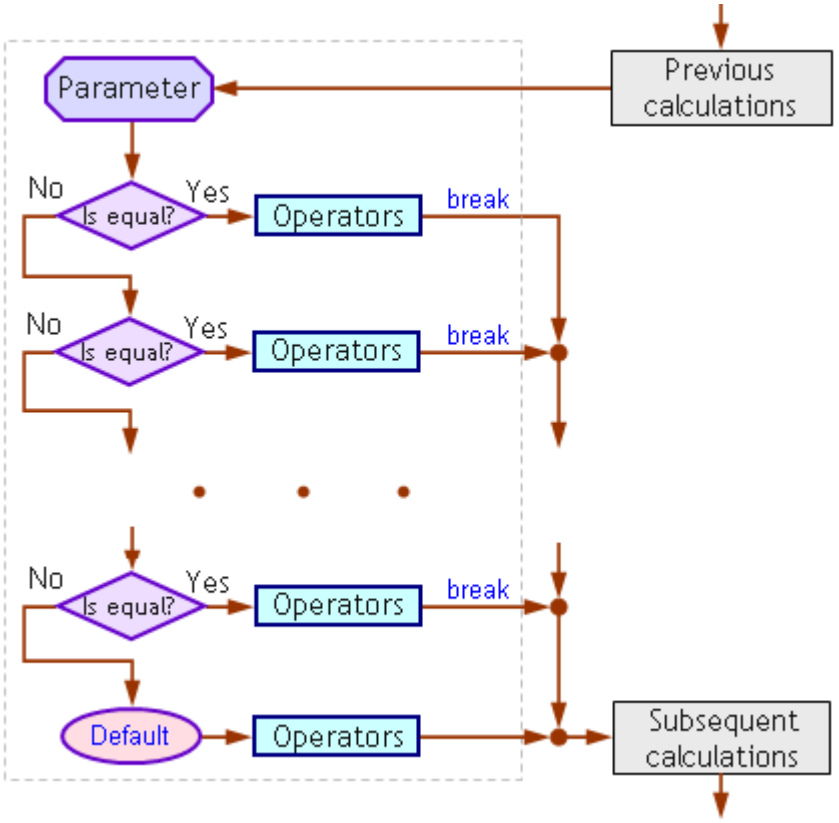
На тази схема се показва работата на оператор switch, както следва. Имаме един параметър / променлива /, който се сравнява с различните случаи /cases/, които ни трябват. При намиране на равенство се изпълняват действията при **ВСИЧКИ ОСТАНАЛИ СЛУЧАИ** след намерения. Ако не бъде намерен такъв случай на нашия параметър, се изпълнява действие по подразбиране или случай с име default. При писане на този оператор нашите възможни случаи не е нужно да са в нарастващ/намаляващ ред.

```
switch(<parameter>) {
    case <value> : <operator>
        ...
        <operator>
    ...
    case <value_n> : <operator>
        ...
        <operator>
    default : <operator>
        ...
        <operator>
}
```

-> запазената дума switch и до нея в скоби параметъра, който ще "изследваме"

-> запазена дума case, последвана от стойност и “ : ” и след тях съответния/те оператор/и
-> default случай, по подобие на обикновените със стойност
Забележка: пред default няма case т.е. case default или default case.

В този вид оператор switch ще изпълни всички вътрешни оператори, които са след намерена стойност и по някой път е неудобно. Затова има вариант това да се спре до кода в самия case. Този вариант е с оператор break за спиране на процеса. Този оператор го пишем в **КРАЯ** на съответния case, където искаме да излезем от оператор switch.



На схемата ясно се вижда как действа switch с break във всеки case, който имаме.

```
switch(<parameter>) {
    case <value> : <operator>
        ...
        <operator>
        break;
    ...
    case <value_n> : <operator>
        ...
        <operator>
        break;
    default : <operator>
        ...
        <operator>
}
```

3. **Задачи**

0. Да се напише програма с помощта на оператор **switch**, която по дадени година и месец изкарва на екрана броя дни в съответния месец.

1. Да се напише програма с помощта на оператор **switch**, която по дадени две числа изкарва по-голямото от тях.

2. Да се напише програма с помощта на оператор **switch**, която по дадени 2 числа и знак за аритметична операция / “+”, “-”, “*”, “/” / да изкара резултата от съответната операция.