**III. Основни видове цикли върху потокови данни. Оператори за цикъл**

**1. Понятие за цикъл**

Често в реалния живот има действия или операции, в които ни се налага да повтаряме повече от веднъж. В програмирането това се отразява чрез така наречените цикли. Цикъл в терминологията на програмните езици представлява код, който може да се изпълнява няколко пъти последователно, в зависимост от някакво условие. Циклите се отличават от останалия код с това, че имат 4 важни елемента:

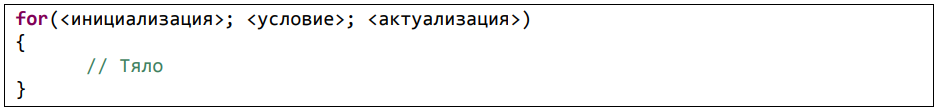
* инициализиране на брояч – използва се най-често, когато броят стъпки е краен и известен. В този елемент се задава началната стойност, за която се извършва цикълът. Понякога се налага този елемент да се дефинира извън самия цикъл. Този брояч може да се използва като условие и за някаква наша употреба, като например да следим итерациите на цикъла.
* условие за изпълнение – всеки цикъл задължително разполага с условие. В зависимост от истинността на това условие, се взима решение дали цикълът да се изпълни още веднъж. Както споменахме по-горе, за това условие може да използваме под някаква форма брояча или някаква друга „булева“.
* актуализация на брояча – тук се задава следващата стойност на брояча. В зависимост от вида на цикъла, тази стъпка отново може да се извърши на различни места.
* тяло – това е код, който ще се изпълнява в цикъла. В зависимост от вида, съществуват два основни вида цикли:
  + цикли с предусловие – цикъл, при който преди изпълнението на тялото се проверява истинността на зададеното условие
  + цикли с постусловие – цикъл, при който проверката за истинността на зададеното условие се проверява след изпълнението на тялото. Циклите от този тип се изпълняват поне веднъж, независимо от това дали условието е изпълнено, или не.

В програмния език С съществуват 3 вида цикли – два от тях са с предусловие и един с постусловие – ***for, while, do-while***. И при трите цикъла е валидно правилото „*докато условието е вярно, до тогава ще се изпълнява тялото“*. **В други езици като C# и java съществува и 4ти вид – foreach. В python съществуват само два - for и while.**

Единичното изпълнение на тялото на цикъл се нарича ***итерация***.

Подобно на if–проверките, нищо не ни пречи да влагаме един цикъл в друг. Тази практика се нарича влагане на цикли а самите цикли – вложени. Често се налага да използваме такива, когато желаем да обходим таблици или други сложни структури. Например, външният цикъл ще обхожда по колони, а вътрешният по редове.

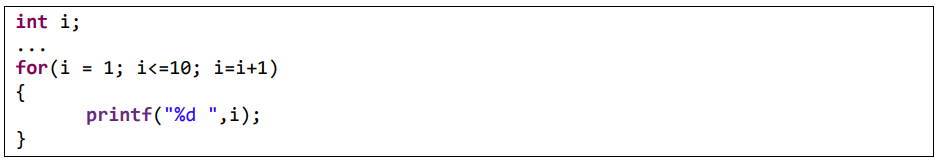
**2. Цикъл FOR**



В този вид цикъл първоначалната инициализация, както и актуализацията, са част от синтаксиса. Цикълът for се употребява, когато броят стъпки е известен. **Този цикъл съществува и в езикa Python, но с по-различен синтаксис – наподобяващ foreach цикъл в C# и Java. СЛЕД АКТУАЛИЗАЦИЯТА НЕ СЛАГАМЕ „;“!**

**Примерна задача**

Да се напише програмен фрагмент, който отпечатва числата от 1 до 10 на екрана.



В показаното решение ясно личат как данните от условието на задачата се прилагат в решението. Създаваме нашия брояч i и го инициализираме със стойност 1. Тази стъпка може да се извърши и в синтаксиса на самия цикъл – for(int i = 1 ; i <= 10; i++) **(В средата Dev-C++ това няма да работи)**. Втората част е да определим нашето условие, тоест i да е по-малко от 10. Накрая актуализираме брояча и пишем тялото.

Изпълнението на програмния фрагмент е следното:

1. Променливата i приема стойност 1.

2. Проверява се истинността на условието i <= 10 (1 <= 10). В този случай условието е „истина“, следователно се преминава към изпълнение на тялото.

3. Отпечатва се текущата стойност на i на екрана.

4. Край на тялото.

5. Преминава се към актуализация – в дадения пример към стойността на i се прибавя единица. Следователно новата стойност на i е 2. **Нищо не пречи и да инкрементираме с повече от 1, това зависи от условието на задачата.**

6. Проверява се истинността на условието за новата стойност i <= 10 (2 <= 10). Преминава се към изпълнение на тялото.

7. Изпълнение на тялото – отпечатване на стойността на i на екрана.

8. Актуализация на i;

... Пропускане на няколко стъпки от изпълнението на програмния фрагмент…

9. След извършване на актуализация i има стойност 11.

10. При проверката на условието i <= 10 (11 <= 10) се установява, че условието не е изпълнено, следователно се напуска изпълнението на цикъла и се изпълнява програмния код след това. След изпълнението на цикъла стойността на i e 11. Това е добре да се знае в случаите, когато се налага употребата на същата променлива.

**В програмния език С e допустимо да се извършат инициализация, проверка и актуализация на повече от една променлива, като това става чрез разделяне със запетая. Пример:**

for (int c = 4, b=3; c < 10 && b < 10; c+=2, b+=3)

{

printf ("%d %d", c, b);

}

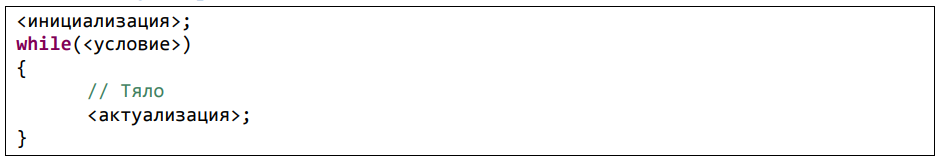
***Важно*: Не е задължително актуализацията или началната стойност на всички променливи да се записват на тези места, те могат да бъдат написани и като изпълними редове някъде в тялото на цикъла.** **Пример за това е безкрайният for – цикъл:**



***Важно: Когато се използва безкраен цикъл, трябва да се предвидят условия за неговото насилствено прекратяване.***

**3. Цикъл WHILE**

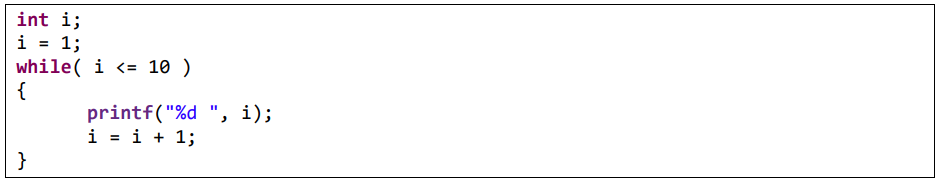
Начин на употреба:



В този вид цикъл, както се вижда от синтаксиса, инициализацията и актуализацията не са вградени в синтаксиса. Това дава свободата за разположението на тези елементи по преценка на автора на програмния код. Присъствието на тези елементи е препоръчително. Цикъл while е отново цикъл с предусловие – условието ще се провери преди изпълнението на тялото.

**Примерна задача**

Да се напише програмен фрагмент, който отпечатва числата от 1 до 10 на екрана.



**Стъпките на изпълнение тук са идентични, както при реализацията с цикъл for.**

**С цикъл while също е възможно създаването на безкрайни цикли, тук идеята е условието да е винаги изпълнено:**

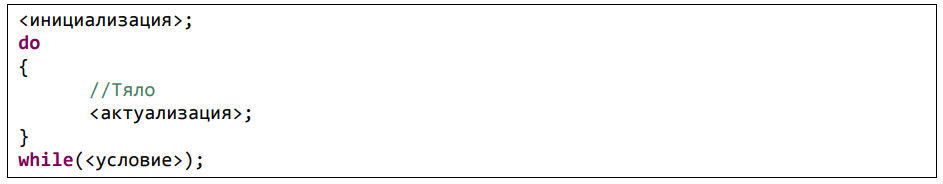


**Горният фрагмент от код е идентичен на while(true). Спомнете си какво говорихме за „булевите“ стойности при проверките. В този тип случай отново трябва да предвидим условие за терминиране на цикъла, тоест да използваме оператора break. Пример по време на часа.**

**Хубаво е да се спомене, че освен break, разполагаме и с оператор continue. Отново по време на часа ще ви покажа употреба, но накратко, когато цикълът стигне continue, се пропуска всичко след него надолу в тялото и се завърта отначало.**

**4. Цикъл DO-WHILE**

Начин на употреба:



Цикълът do-while е единственият в програмния език С, който е с постусловие. **В този случай, тялото на цикъла ще се изпълни поне веднъж, независимо от това дали е изпълнено условието.** Причината за това е, че условието се намира след тялото в кода. При do-while също частите за инициализация и актуализация не са предвидени в синтаксиса.

**Примерна задача**

Да се напише програмен фрагмент, който отпечатва числата от 1 до 10 на екрана.





Съкратеното записване на някои оператори спомага за по-четлив код, както и работи по-бързо от нормалния запис. Има особености при времето на изпълнение на действието, като трябва да се внимава при комбинации между отделните оператори.

**5. Задачи за упражнение**

1. Да се напише програма, която прочита от клавиатурата едно цяло число и изкарва числова пирамида:

1

2 2

3 3 3

.....

1. Да се напише програма, която прочита от клавиатурата две цели числа и извежда сумата на числата между тези две числа. Въведените числа от клавиатурата не е задължително да са в ред по-малко, по-голямо.

Вход: 2 7

Изход: 18 Обяснение 3+4+5+6 = 18

1. Да се напише програма, която прочита от клавиатурата две числа и извежда сумата на четните числа и произведението на нечетните, които се намират между тези две числа.
2. Да се напише програма, при която се въвеждат две числа N и К. След това се въвеждат още N числа и се извежда броят на числата, които са по-големи от K и се делят на 3.
3. Да се напише програма, която при въвеждането на произволен брой числа извежда сборът им. За край на въвеждането се приема въвеждането на 0.
4. Да се напише програма, която принтира ASCII стойностите на всички малки латински букви.