Сразу разберём пример. Здесь мы выводим на экран числа от 0 до 99:

*// Заведем переменную i целого типа и равную нулю*

**int** i = 0;

*// знак цикла while означает*

*// «повторяй блок кода { … } пока верно условие внутри скобок», то есть (i < 100)*

**while** (i < 100) {

System.out.println (i); *// распечатать текущее значение i*

i = i + 1; *// увеличить i на 1*

}

System.out.println ("Жизнь за циклом продолжается");

Когда компилятор встретит слово **while**, он сразу поймёт, что это цикл. Код внутри фигурных скобок после **while**будет выполняться всё время, пока верно условие в круглых скобках. В данном случае в круглых скобках написано**(i < 100)**, то есть мы ограничиваем переменную **i** значением 100. Поэтому блок кода внутри фигурных скобок повторится 100 раз.  
  
В фигурных скобках мы написали – распечатать**i**, затем увеличить её на 1. Этот код будет выполнен много раз, и каждый раз переменная **i** будет увеличиваться. Сначала она равна 0, потом 1, потом 2 и так далее, и наконец она станет равной 99. Когда **i** равно 100, то условие **i < 100** перестает выполняться, и цикл заканчивается. Заметьте, что поскольку мы написали i строго меньше 100, то последнее значение **i=100** в цикл уже не входит. Как часто делают программисты, мы начали с нуля, а не с 1. Посчитайте: от 1 до 99 будет 99 чисел, а от 0 до 99 получится ровно 100 чисел. Поэтому наш цикл сделает 100 оборотов.  
  
В данном случае мы сами решили начать с нуля и поставить условие **(i<100)**. Мы могли бы начать и с 10 и поставить, например, **i <=100**. Читается такое условие "**i** меньше либо равно 100". То есть теперь уже 100 включено в последний оборот цикла. В будущих уроках мы будем проходить массивы – наборы чисел, которые нумеруются с нуля. Тогда начинать с нуля и заканчивать строгим неравенством будет необходимо. Пока что вы можете начинать с чего угодно и заканчивать как угодно, лишь бы программа делала в точности то, чего вы от неё хотите.  
  
Переменные подобные **i**, которые постоянно меняют свои значения в цикле, часто называют счетчиками. Это обиходное, неофициальное название, тем не менее оно очень удобно – ведь с помощью переменной **i** мы можем сразу посчитать, на каком обороте цикла мы находимся.

В общем виде цикл **while**выглядит так:

**while (условие) { тело цикла }  
т.е. повторять тело цикла, пока верно условие**

**Задача 1.**Как сделать так, чтобы этот код печатал числа не от 0 до 99, а от 50 до 201?

**int** i = 0;

**while** (i < 100) {

System.out.println (i);

i = i + 1;

}

**Задача 2.**Что будет напечатано по итогу выполнения этого кода и почему?

**int** x = 10;

**int** y = 100;

**for** ( ; x < y; x += 20, y --) ;

System.out.println ("x равен " + x + " ; y равен " + y);

**Правильный вывод программы**

x равен 110 ; y равен 95

**Задача 3.** Посчитаем сумму чисел 1, 2, …, 9.

**Задача 4.** Посчитаем сумму нескольких чётных чисел 8+10+12+14.

**Задача 5.** Посчитаем произведение 4\*8\*12\*16\*20. То есть нужно все эти числа перемножить между собой.

**Задача 6.** Посчитаем выражение 1-2+3-4+5-6.

**Задача 7.** Посчитаем значения последовательности чисел по такому правилу:  
a1 = 5,  
a2 = a1\*3 + 1,  
a3 = a2\*3 + 1,  
…  
a100 = a99\*3 + 1.  
  
Если a100 больше 2000000000000000000l, то вывести на экран фразу «до фига получилось!» :) Также вывести количество пройденных оборотов цикла.  
Такие последовательности будут нужны, если вас интересует набор чисел, заданных любым соотношением и следующее вычисляется из предыдущего по известному правилу.

**Задача 8.** Зададим вопрос и будем переспрашивать несколько раз, пока человек не ответит правильно. Важно понимать – вы можете заставить программу повторить любой набор действий много раз, а не только вычислять что-либо в цикле.

**Задача 9.**Напишите калькулятор. Предложите пользователю ввести число, затем знак одной из арифметических операций **+**,**-**,**\***,**/**. Введите ещё число, выполните операцию, распечатайте результат и предложите ввести новую. Если пользователь введёт **q**, то остановите программу.  
  
Вариант работы программы:  
Остановка программы буквой q  
Введите число  
10  
Введите операцию  
-  
Введите число  
100  
Результат -90  
Введите операцию  
\*  
Введите число  
2  
Результат -180  
Введите число  
q  
Окончательный результат -180

**Двойные и тройные вложенные циклы**

Сам по себе цикл означает, что какой-то блок кода будет повторяться. Внутри этого блока кода можно заставить повторяться другой блок кода, т.е. один цикл можно писать внутри другого. Такие циклы называются вложенными. Они нужны, например, для работы с любыми таблицами.

Рассмотрим распечатку таблицы умножения:

System.out.println ("Это таблица умножения!");

**for** (**int** a = 1; a < 10; a ++) {

*//внутри первого цикла можно сделать 2ой*

**for** (**int** b = 1; b < 10; b ++)

System.out.print (" " + a\*b); *//печатаем числа через пробел*

System.out.println (); *//поставить enter для новой строчки*

}

Вывод программы:  
1 2 3 4 5 6 7 8 9  
2 4 6 8 10 12 14 16 18  
3 6 9 12 15 18 21 24 27  
4 8 12 16 20 24 28 32 36  
5 10 15 20 25 30 35 40 45  
6 12 18 24 30 36 42 48 54  
7 14 21 28 35 42 49 56 63  
8 16 24 32 40 48 56 64 72  
9 18 27 36 45 54 63 72 81  
  
Чтобы распечатать одну строчку нужен цикл. А у нас тут целый набор строк. Для перебора всех строк и нужен второй цикл. Он ставит enter после каждой строки, и знает номер текущей строки.  
  
Компилятор выполняет все команды подряд. Он идет по коду сверху вниз и встречает первый **for.**Для него это значит - выполнить тело этого цикла столько-то раз. Он начинает выполнять цикл первый раз, снова сверху вниз и встречает вложенный цикл. Снова это значит - выполнить тело этого цикла столько-то раз. Выполнив весь вложенный цикл, компилятор идёт дальше, а дальше - новый оборот первого цикла.  
  
Так получаем, что на каждом обороте главного верхнего цикла будет полностью выполнены все обороты вложенного цикла. В нашем примере это означает, что на одном обороте главного цикла будет напечатана целая строка из таблицы умножения.  
  
Подобная ситуация часто возникает в играх с клетками на игровом поле по типу морского боя. Вообще с любыми таблицами так или иначе приходится отсчитывать номер строки и номер ячейки внутри этой строки, и очень помогает двойной цикл. Мы напишем игру "морской бой" в разделе 5 про массивы и таблицы.

**Задача 10.**Распечатать символ a, так что в первой строке всего один символ, во второй - два, в третьей - три, и так далее до 5.

*/\* Будет вот так:*

*a*

*aa*

*aaa*

*aaaa*

*aaaaa \*/*

**Задача 11.**Ниже дан код, печатающий "пирамидку" из букв 'a', как было в материале выше. Как изменить код, чтобы печатался только контур этой пирамидки, а вместо всех внутренних символов стояли бы пробелы?

*/\* Сейчас печатается так:*

*a*

*aa*

*aaa*

*aaaa*

*aaaaa \*/*

*/\* Как сделать, чтобы печаталось так:*

*a*

*a*

*a*

*a*

*a \*/*

**Задача 12.**Распечатайте на экран вопрос, "Сколько будет 4\*5?". Введите число с ответом клавиатуры. Если пользователь ответил правильно, то поздравьте его и завершите программу. Если же пользователь ответил неверно, то переспросите его, введите новое число, снова проверьте на правильность ответ и так далее, пока не ответит правильно.  
  
Важно понимать – вы можете заставить программу повторить любой набор действий много раз, а не только вычислять что-либо в цикле.

**Задача 13.**Распечатайте в программе таблицу умножения чисел от 1 до 10. На экране должно получиться:  
  
1 2 3 4 5 6 7 8 9  
2 4 6 8 10 12 14 16 18  
3 6 9 12 15 18 21 24 27  
4 8 12 16 20 24 28 32 36  
5 10 15 20 25 30 35 40 45  
6 12 18 24 30 36 42 48 54  
7 14 21 28 35 42 49 56 63  
8 16 24 32 40 48 56 64 72  
9 18 27 36 45 54 63 72 81

**Задача 14.**Вычислить последовательно числа и все вывести на экран:

a1 = 10

a2 = 10 + a1

a3 = 10 + a2

....

a100 = 10 + a99

**Вывод программы**

В консоли это будет выглядеть примерно так:  
Это a1 и она равна 10  
Это a2 и она равна 20  
Это a3 и она равна 30  
...

**Задача 15.**Вычислите последовательно числа (даны ниже) и выведите их на экран, и также посчитайте их сумму. То есть конкретно посчитайте a1+a2+a3+...+a10  
  
Уточните, что результат укладывается в максимальное значение выбранного вами типа данных и что дробная часть нигде не отброшена.  
  
Имейте в виду, что a1, a2, a3 нужно не только вычислить по отдельности, но ещё и сложить их всех между собой в какой-то отдельной переменной, чтобы посчитать сумму. Значение результата может не поместиться в double - оцените, начиная с какого номера a становится Infinity. Infinity обозначает бесконечность и это уже не обычное число, это значит, что программа работает не нормально. Старайтесь исправлять ситуации в боевых ситуациях, чтобы не приходилось иметь дело с такими огромными числами, а пока просто познакомимся с ними.

a1 = 5

a2 = sqrt ( 10+ a1\*a1\*a1/1000000 ) + 2\* a1\*a1

a3 = sqrt ( 10+ a2\*a2\*a2/1000000 ) + 2\* a2\*a2

...

a10 = sqrt ( 10+ a99\*a99\*a99/1000000 ) + 2\* a99\*a99

**Вывод программы**

В консоли это выглядит так:  
  
Это a1 и оно равно 53.16229742434199  
При этом сумма равна 53.16229742434199  
Это a2 и оно равно 5655.645680390971  
При этом сумма равна 5708.807977815313  
Это a3 и оно равно 6.397308146333617E7  
При этом сумма равна 6.397879027131399E7  
Это a4 и оно равно 8.185110815506304E15  
При этом сумма равна 8.185110879485094E15  
Это a5 и оно равно 1.3399207812497706E32  
При этом сумма равна 1.3399207812497706E32  
Это a6 и оно равно 3.590775400049991E64  
При этом сумма равна 3.590775400049991E64  
Это a7 и оно равно 2.5787335947208342E129  
При этом сумма равна 2.5787335947208342E129  
Это a8 и оно равно Infinity  
При этом сумма равна Infinity  
Это a9 и оно равно Infinity  
При этом сумма равна Infinity  
Это a10 и оно равно Infinity  
При этом сумма равна Infinity

#### **Задача №1:**

Необходимо вывести на экран числа от 1 до 7. То есть на экране должно быть: 1 2 3 4 5 6 7

#### **Задача №2:**

А если бы мы написали код вот так, то **код** **бы не выполнился ни разу.**

class Test {

    public static void main(String args[]) {

        int i = 1;

        while (i < 0) {

            System.out.print(i + " ");

            i++;

        }

    }

}

Почему?

#### **Задача №3:**

А если бы код был так написан вот так, то на экран бы выводились и выводились до бесконечности числа 1 2 3 4 5 6 7 ….

class Test {

    public static void main(String [args]) {

        int i = 1;

        while (true) {

            System.out.print(i + " ");

            i++;

        }

    }

}

#### **Задача №4**

Необходимо вывести на экран вот такую последовательность чисел:

100 90 80 70 60 50 40 30 20 10

#### **Задача №5**

Необходимо вывести на экран числа от 3 до -3. То есть на экране должно быть: 3 2 1 0 -1 -2 -3