Курс: «Разработка программного обеспечения на Java»

Тема: Циклы

План

1. Что такое цикл? Необходимость использования циклов
2. Цикл while
3. Цикл do-while
4. Цикл for
5. Операторы break и continue
6. Вложенные циклы. Примеры использования

1. Необходимость использования циклов

Циклом называется любая многократно исполняемая последовательность инструкций. Каждое выполнение такой инструкции называется итерацией. Для того, чтобы определить, будет ли в очередной раз выполняться итерация или цикл завершится, используется условное выражение. Если результат выражения – истинное значение, тогда тело цикла будет выполнено, если нет, тогда цикл будет завершен.

Существуют несколько видов циклов, которые определяют тип проверки условия (до или после выполнения блока цикла), а также, цикл с счетчиком и цикл с упрощенной записью.

Примеры использования циклов

Стоит задача посчитать сумму первых десяти нечетных чисел. Если не использовать циклы, мы получим большое количество однотипного кода и необходимо будет самостоятельно вычислять входные данные.

Получим сумму без цикла:
int sum = 1 + 3 + 5 + 7 + 9 +11+ 13 + 15 + 17 + 19;

Выглядет компактно, но что будет, если в условии заменить десять чисел на тысячу или на сто тысяч? Тогда не обойтись без использования цикла.

Запишем схематически:
```
int sum = 0, i = 1;
цикл<условие>{
    sum += i;
    i+=2;
}
```

В данном случае условие определяет количество итераций, которых необходимо для получения первых N нечетных чисел, i - следующее нечетное число.

Как видно из примера, использование циклов является более удобным и универсальным способом многократного исполнения однотипного набора инструкций.

2. Цикл while

Цикл while выполняется, пока истинно условие, которое указанное перед его началом. Это условие проверяется до выполнения тела цикла, поэтому тело может быть не выполнено ни разу (если условие с самого начала ложно).

Синтаксис:

```
while (<условие>) {
  <тело цикла>
}
```

Пример

Задача: Вычислить сумму чисел от 1 до n.

Решение:

```
int i = 1, sum = 0, n = 10;
while (i < n) {
  sum += i++;
}
```

3. Цикл do while

В цикле с постусловием (do while) условие проверяется после выполнения тела цикла. Отсюда следует, что тело выполнится хотя бы один раз.

Пример

Задача: Определить, является ли число n простым.

Решение:

```
int i = 2; n = 7;
boolean isPrime = true; // считаем, что число простое
do {
   if (n%i==0) {
            isPrime = false; // если число делится без остатка не только на само себя и единицу, тогда считаем число составным
   }
} while (i++<n)
```

4. Цикл for

Цикл for – это цикл со счётчиком. В нем некоторая переменная изменяет своё значение от заданного начального значения до конечного значения с некоторым шагом, и для каждого значения этой переменной тело цикла выполняется один раз.

Синтаксис:
for (<инициализация счетчика>;<условие выполнения>;<выполняемая операция после каждой итерации>){
  <тело цикла>
}

Пример

Задача: Определить факториал числа n.

Решение:

```
int f = 1; n = 5;
for(int i = 2; i <= n; i++){
    f *= i;
}
```

5. Операторы break и continue

В языке Java существуют три оператора перехода: break, continue и return. Операторы break и continue применяются в циклах в том случае, если необходимо преждевременно завершить выполнения цикла или конкретной итерации.

Рассмотрим оператор выхода из цикла. Оператор **break** позволяет спровоцировать немедленное завершение работы цикла, программа передаст управление следующему за циклом оператору.

Рассмотрим оператор прерывания итерации. Оператор **continue** позволяет прервать текущую итерацию (при этом проигнорировав часть операторов внутри тела цикла) и перейти к следующей итерации, если условие выполнения цикла будет истинным.

Пример использования оператора break

Необходимо вычислить сумму соседних натуральных чисел (на промежутке от 1 до 1000), произведение которых равняется n.

```
int sum = 0, n = 90;
for(int i = 1; i < 1000; i++){
    if(i*(i+1) == n)
            sum = i+i+1;
            break;
}

if(sum > 0)
     System.out.println(sum); // Если сумма больше нуля (значения по умолчанию), тогда выведем результат на экран
```

Пример использования оператора continue

Вывести в ряд все числа, которые не кратны 4 и 6, на промежутке от 0 до 1000.

```
for(int i = 0; i <= 1000; i++){
    if(i % 4 == 0 || i % 6 == 0)
                continue;
    System.out.print(i);
}
```

6. Вложенные циклы. Примеры использования

Циклы способны многократно повторять выполнение не только других операторов, но и повторять несколько раз выполнение других циклов. Цикл, который многократно выполняется внутри другого цикла называется вложенным циклом.

При использовании вложенных циклах количество повторений перемножается. Например, если в цикл из 10 повторений вложить цикл из 5 повторений, то последний выполнится 50 раз.

Используйте вложенные циклы для организации более сложных вычислительных методов, перебора двумерных массивов, повторения блока операторов с различными входными данными.

Пример использования вложенных циклов

Выведем в консоль треугольник из символов "*".

```
for (int i = 0; i < 15; i++) {
        for (int j = i; j < 15; j++) {
            System.out.print("*");
        }
        System.out.println();
}
```

Вложенный цикл выводит в ряд некоторое количество символов. Каждое следующее выполнение вложенного цикла будет начинаться со значения счетчика i внешнего цикла, т.е. на единицу меньше. В результате получим фигуру из символов:

```
**********
*********
********
*******
******
*****
****
***
**
*
```