Looping

Johannes Alexander Putra

Materials

- Loops
- While Loop
- Do While Loop
- For Loop
- Nested Loop

Part one

Loop definition
While Loop
Do While Loop
For Loop

Loops

With a loop, a part of a program is repeated over and over,until a specific goal is reached.

Two types of loops in computer programming:

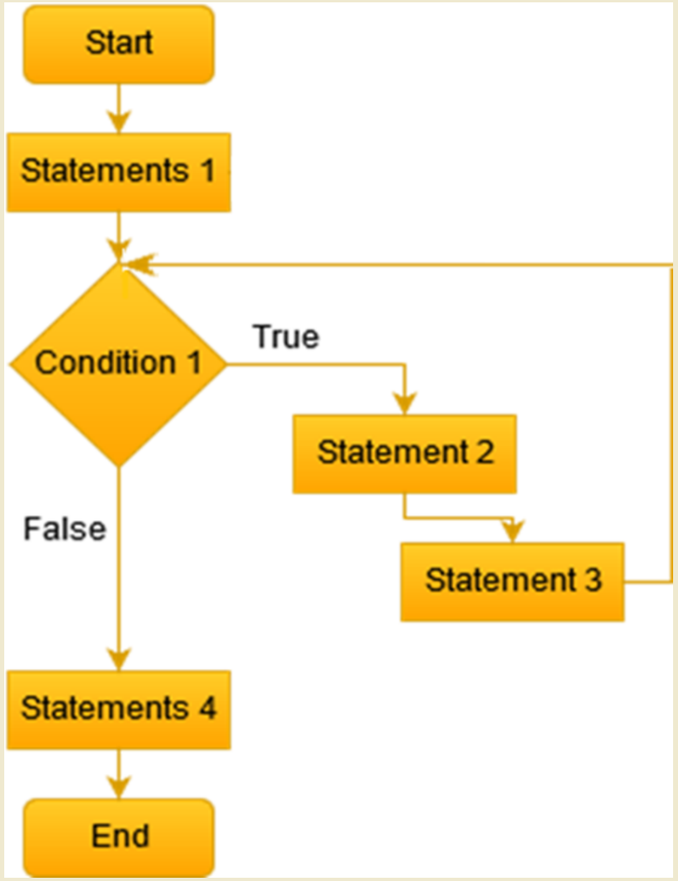
- You know exactly how many times you have to repeat
- You know the specific condition when to stop the repetition, but you do not know how many times to repeat the action to achieve that particular condition

While Loop

- As long as the condition is true, the statements will always be repeated.
- while is the best choice to perform repetition when it does not know how many times it should be repeated (but do know what the condition that should make the repetition to stop)

```
while(conditions)
{
   statement;
   statement;
   statement;
   ...
}
```

While Loop




```
public static void main(String[] args){
  int result = 0;
  Scanner sc = new Scanner(System.in);
  int num1 = sc.nextInt();
  int num2 = sc.nextInt();
  while(num1>=num2){
    num1-=num2;
    result+=1;
  }
  System.out.println(result);
}
```

While More Example


```
public static void main(String[] args){
  String result = "";
  int i = 1;
  while(i<=10){
    result+=i+"";
    i+=2;
  }
  System.out.println(result);
}
```

What is output of this syntax?

```
public static void main(String[] args){
    int res = 2;

   while(res%2!=0||res%3!=0||res%4!=0||res%5!=0){
      res+=1;
   }
   System.out.println(res);
}
```

Do While Loop

- The statements inside the block will always be repeated as long as the condition is true.
- The do-while loop is appropriate when the loop body must be executed at least once.

```
do{
   statement;
   statement;
   statement;
   ...
}while(conditions);
```

Do While Loop

```
public static void main(String[] args){
   Scanner sc = new Scanner(System.in);
   int pilihan, a, b;
   do{
      a = sc.nextInt();
      b = sc.nextInt();
      pilihan = sc.nextInt();
      if(pilihan==1){
        System.out.println(a-b);
      }
      else if(pilihan==2){
          System.out.println(a+b);
      }
}while(pilihan==1||pilihan==2);
}
```

For Loop

```
for(initialization;condition;updater){
  statement;
  statement;
  statement;
}
```

- initialization: one or more statements which are executed only one time in the beginning of the iteration.The counter/iterator usually initialize in the initialization
- condition: decide whether the block statement will be executed or not. If the expression returns false, then the whole for statement is terminated and the program will evaluate the next statement.
- updater: statement which is executed after the block statement is executed. The counter/iterator's value usually updated in the updater

For Loop

```
public static void main(String[] args){
   int n = 3;
   String result = "“;
   for(int i = 1; i<=n; i++){
         System.out.print(i+" ");
   }
System.out.println();
}
```

Part two

Nested Loop
Break & Continue

Nested Loop Case Example

- Here, we are using a for loop inside another for loop.
- We can use the nested loop to iterate through each day of a week for 3 weeks.
- In this case, we can create a loop to iterate three times (3 weeks). And, inside the loop, we can create another loop to iterate 7 times (7 days).

```java
public class Main {

  public static void main(String[] args) {

    int weeks = 3;
    int days = 7;

    // outer loop prints weeks
    for (int i = 1; i <= weeks; ++i) {
      System.out.println("Week: " + i);

      // inner loop prints days
      for (int j = 1; j <= days; ++j) {
        System.out.println("  Day: " + j);
      }
    }
  }
}
```

for loop inside the while loop

We can also create nested loops with while and do...while in a similar way.

```java
public class Main {
  public static void main(String[] args) {

    int weeks = 3;
    int days = 7;
    int i = 1;

    // outer loop
    while (i <= weeks) {
      System.out.println("Week: " + i);

      // inner loop
      for (int j = 1; j <= days; ++j) {
        System.out.println("  Days: " + j);
      }
      ++i;
    }
  }
}
```

Java nested loops to create a pattern

We can use the nested loop in Java to create patterns like full pyramid, half pyramid, inverted pyramid, and so on. Here is a program to create a half pyramid pattern using nested loops.

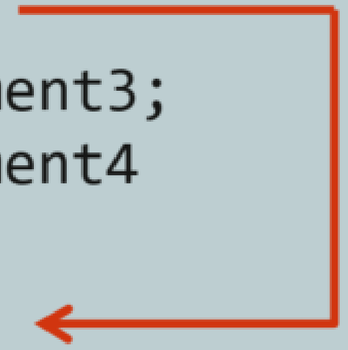
```java
public class Main {
  public static void main(String[] args) {

    int rows = 5;

    // outer loop
    for (int i = 1; i <= rows; ++i) {

      // inner loop to print the numbers
      for (int j = 1; j <= i; ++j) {
        System.out.print(j + " ");
      }
      System.out.println("");
    }
  }
}
```

Break

The if break statement is executed within the loop. the loop statement should be terminated immediately

the program will continue running with the statement after the loop

```
while(condition){
    statement1;
    statement2;
    break;
    statement3;
    statement4
}
statement;  [statement outside the while loop]
```



Kontrol diteruskan ke pernyataan di luar loop

Use Break Inside While Loop

- Output: `0 1 2 3`
  - Loop berhenti dijalankan jika penghitung loop sama dengan 4

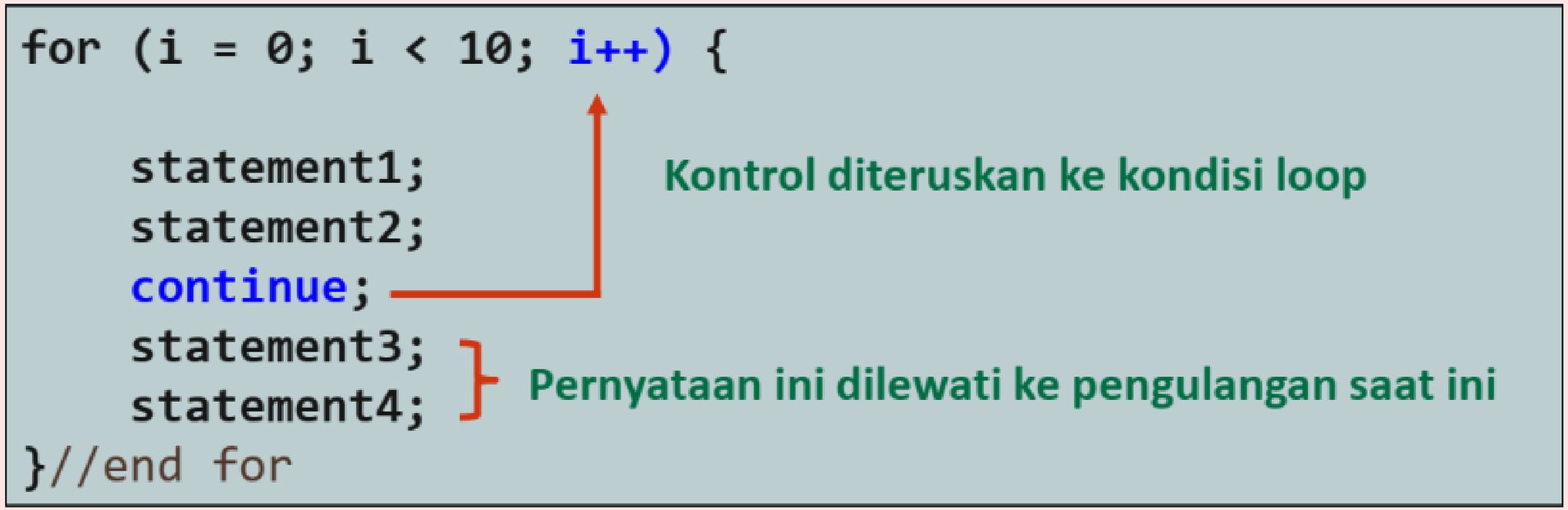
```
public static void main(String[] args) {
    int i = 0;
    while (i < 10) {
        System.out.println(i + "\t");
        i++;
        if (i == 4) {
            break;
        }//endif
    }//end while
}//end method main
```

Continue

sometimes you may want to skip the current loop in the loop and not terminate the loop itself

You can use the continue statement to skip the current loop in the loop

- that is, the remainder of the loop is skipped to the end of the loop. However this doesn't end the loop
- when the program reaches the end of the loop it will again test the loop continuation condition


for (i = 0; i < 10; i++) {
statement1;
statement2;
continue;
statement3;
statement4;
}//end for
Kontrol diteruskan ke kondisi loop
Pernyataan ini dilewati ke pengulangan saat ini

- Output: `0 1 2 3 5 6 7 8 9`
  - Output tidak meliputi 4
  - Karena pernyataan continue, eksekusi loop dilewati saat penghitung loop adalah 4

```
public static void main(String[] args) {
   for (int i = 0; i < 10; i++) {
      if (i == 4) {
         continue; //control jumps to update i++
      }//endif
      System.out.print(i + "\t");
   }//end for
}//end method main
```

Finish

Continue by Excercises