



ORACLE

Academy



Java Foundations

6-3

Using break and continue Statements

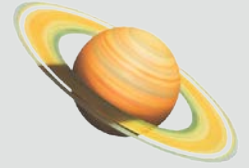
ORACLE
Academy



Objectives

- This lesson covers the following objectives:
 - Use a break statement to exit a loop
 - Use a continue statement to skip part of a loop
 - Explain the need for loop comments





Mission to Saturn's Rings

- Let's consider another scenario for this mission
 - As the spaceship is rotating around Saturn and taking snapshots, the robotic arm or camera breaks
- How would you solve this problem?
 - If you were to write a Java program, which construct would you use?
 - Let's see whether Java has a statement that enables you to end a loop immediately

How Do You Exit a Loop Early?

- Usually, the only way to exit a loop is for the loop condition to evaluate to false
- However, it's often convenient to terminate a loop early when certain conditions are met
- In such cases, continuing to loop would be a waste of processor time

How Do You Exit a Loop Early?

- You can use two Java statements to terminate a loop early:
 - **break**
 - **continue**

Using break in a Loop

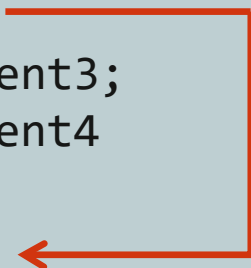
- When a break statement is executed inside a loop, the loop statement is terminated immediately
- The program continues to execute with the statement following the loop statement
- Syntax:

```
break;
```

Using break in a while Loop

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

Control passes to the statement outside the loop

A red line originates from the 'break;' statement inside the while loop, extends horizontally to the right, then turns 90 degrees downward, and finally turns 90 degrees left to point at the 'statement;' line outside the loop. This visualizes the immediate exit from the loop upon encountering the break statement.

Using break in a while Loop: Example

- Output:

0	1	2	3
---	---	---	---

 - Execution of the loop is terminated when the loop counter is equal to 4

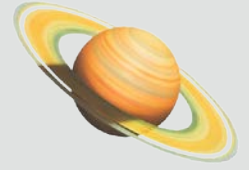
```
public static void main(String[] args) {  
    int i = 0;  
    while (i < 10) {  
        System.out.println(i + "\t");  
        i++;  
        if (i == 4) {  
            break;  
        }//endif  
    }//end while  
}//end method main
```

Using break in a for Loop

- Let's write a program to demonstrate a break statement in a for loop
- The program must ...
 - Read 10 numbers from the console
 - Compute the sum of the numbers that the user enters
 - If the user enters 999, terminate the loop regardless of the value of the loop counter and without adding to the sum

Using break in a for Loop: Example

```
public static void main(String[] args) {  
    Scanner in = new Scanner(System.in);  
    int numInputs = 10, input = 0, sum = 0, stopLoop = 999;  
    System.out.println("Enter 10 numbers");  
    for (int i = 0; i < numInputs; i++) {  
        input = in.nextInt();  
        if (input == stopLoop){  
            break;  
        }  
        else {  
            sum += input;  
        }  
    }  
    System.out.println("The sum of the numbers:" + sum);  
}
```



Mission to Saturn's Rings: Implementing the Conditions

- Let's use a while loop and a break statement to implement the conditions specified at the beginning of the lesson

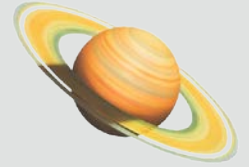
```
public static void main(String[] args) {  
    long distTravelled = 0;  
    long maxDistance = 50000000;  
  
    while (distTravelled <= maxDistance) {  
        if (isCameraBroken()) {  
            break;  
        }  
        else {  
            cameraSnap();  
        }  
    }  
    shipRotate();  
} //end method main
```



Exercise 1

- Create a new project and add the `ComputeSum.java` file to the project
- Examine `ComputeSum.java`
- Implement the following:
 - Accept 10 numbers from the user
 - Compute the sum of the numbers entered
 - When 0 is entered, the program must exit and display the sum of the numbers

Mission to Saturn's Rings: Another Scenario



- Let's consider another scenario for this mission
 - As the spaceship is rotating around Saturn and taking snapshots of Saturn's rings ...
 - If the visibility is zero, do not take snapshots
 - Otherwise, continue to take the snapshots
- How would you solve this problem?
 - If you were to write a Java program, which construct would you use?
 - Let's see whether Java has a statement that enables you to skip the current iteration of the loop

Using continue in a Loop

- Sometimes, you may want to skip the current iteration in a loop and not terminate the loop itself
- You can use a continue statement to skip the current iteration in a loop:
 - That is, the rest of the loop body is skipped to the end of the loop
 - However, it doesn't end the loop
 - When the program reaches the end of the loop, the program jumps back to test the loop continuation condition
- Syntax:

```
continue;
```

Using continue in a while Loop

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

Control passes to the loop condition

These statements are skipped in the current iteration

Using continue in a for Loop

```
for (i = 0; i < 10; i++) {
```

```
    statement1;
```

```
    statement2;
```

```
    continue;
```

```
    statement3;
```

```
    statement4;
```

```
} //end for
```

Control passes to the loop condition



These statements are skipped in the current iteration



Using continue in a for Loop

- Output: `0 1 2 3 5 6 7 8 9`
 - The output doesn't include 4
 - Because of the continue statement, the loop execution is skipped when the loop counter is 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
```

Putting It All Together

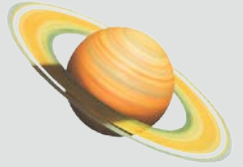
- Let's write a program using the while loop and the continue statement
- The program must ...
 - Compute the sum of numbers between 1 and 99 using the while loop
 - If the number is a multiple of 10, the current iteration must be skipped and the number must not be added to the sum
 - Display the sum to the console



Computing the Sum of Numbers

```
public static void main(String[] args) {  
    int counter = 0;  
    int sum = 0;  
    while (counter < 100) {  
        counter++;  
        if (counter % 10 == 0) {  
            continue;  
        }  
        else {  
            sum += counter;  
        }  
    }  
    System.out.println("Sum of 1 - 99: " + sum);  
}
```

Is this a multiple of 10? If yes, then skip the current iteration



Mission to Saturn's Rings: Implementing the Conditions

- Let's use a `while` loop and a `continue` statement to implement the conditions specified at the beginning of this topic

```
public static void main(String[] args) {  
    long distTravelled = 0;  
    long maxDistance=50000000;  
  
    while (distTravelled <= maxDistance) {  
        if (getVisibility() == 0) {  
            continue;  
        }  
        else {  
            cameraSnap();  
        }  
    }  
    shipRotate();  
}
```

Exercise 2

- Add the file `CountChar.java` to the project you created for exercise 1
- Examine `CountChar.java`
 - The program is used to count the number of occurrences of the char 'w' in the string
 - Modify the program to ...
 - Resolve the syntax error
 - Print the count of char 'w'
 - Expected Output:
 - Number of w : 3



Exercise 3

- Add the file `BreakContinue.java` to the project you created for exercise 1
- Examine `BreakContinue.java`
- Modify the program by using `break` and `continue` statements ...
 - If the number is even, the number should not be printed
 - Execution of the loop should stop when the value of the loop counter is 7

Writing Loop Comments

- It's a good practice to add appropriate comments to loops
- Otherwise ...
 - Code tends to be confusing to look at
 - You won't be able to understand the logic very easily
- It helps to understand ...
 - Loop variables used and their purpose
 - Logic of the loop
 - Number of iterations
 - Execution of the statements in the loop depending on the condition or criteria or both

Writing Loop Comments: Example

```
public static void main(String[] args) {  
  
    Scanner in = new Scanner(System.in);  
    int numInputs = 10, input = 0;  
  
    //This loop is executed 10 times  
    for (int i = 0; i < numInputs; i++) {  
        input = in.nextInt(); //user inputs a number  
  
        if (input % 2 == 0) { //if the number is even skip the  
            continue;        //remaining code and restart the loop  
        } //endif  
  
        System.out.println("That number was odd");  
    } //end for  
} //end method main
```

Exercise 4

- Add the file `Divisors.java` to the project you created for exercise 1
- Examine `Divisors.java`
- The program finds all divisors of a number

Exercise 4

- Modify the program to include comments for the loop about ...
 - Loop variables used
 - Logic of the loop
 - Number of iterations
 - Condition used
 - Control flow in the loop

Summary

- In this lesson, you should have learned how to:
 - Use a break statement to exit a loop
 - Use a continue statement to skip part of a loop
 - Explain the need for loop comments





ORACLE

Academy

