How to prevent infinite loops in java programming ?

To prevent infinite loops , we can use

1.break

2.System.exit()

Break

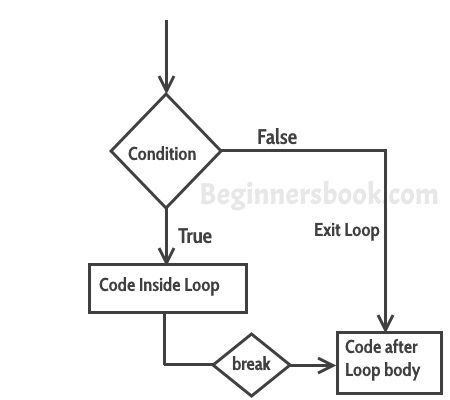
When a break statement is encountered inside a loop, the loop is immediately terminated and the program control resumes at the next statement following the loop.

We can use Java break statement in all types of loops such as [for loop](https://www.javatpoint.com/java-for-loop), [while loop](https://www.javatpoint.com/java-while-loop) and [do-while loop](https://www.javatpoint.com/java-do-while-loop).

Break statement is almost always used with decision-making statements ([Java if...else Statement](https://www.programiz.com/java-programming/if-else-statement)).

Here is the syntax of the break statement in Java:

break;



# System.exit() in Java

The **java.lang.System.exit()** method exits current program by terminating running Java virtual machine. This method takes a status code. A non-zero value of status code is generally used to indicate abnormal termination. This is similar [exit in C/C++](https://www.geeksforgeeks.org/understanding-exit-abort-and-assert/).

Following is the declaration for **java.lang.System.exit()** method:

public static void exit(int status)

**exit(0)** : Generally used to indicate successful termination.  
**exit(1) or exit(-1) or any other non-zero value** – Generally indicates unsuccessful termination.

**Note :**This method does not return any value.

# Note : System.exit() in Java It is the system call but not the language statement

# Example:

|  |  |  |
| --- | --- | --- |
| class  {      public static void main(String[] args)      {          int arr[] = {1, 2, 3, 4, 5, 6, 7, 8};            for (int i = 0; i < arr.length; i++)          {              if (arr[i] >= 5)              {                  System.out.println("exit...");                    // Terminate JVM                  System.exit(0);              }              else                  System.out.println("arr["+i+"] = " +                                    arr[i]);          }          System.out.println("End of Program");      }  } |  | Output:  arr[0] = 1  arr[1] = 2  arr[2] = 3  arr[3] = 4  exit... |

# Difference between break and java

|  |  |
| --- | --- |
| break | Exit() |
| Break is a keyword | Exit() is standard library function |
| Break causes an immediate exit from switch or loop(for, while or do). | Exit() terminates complete program execution when it is called. |
| Break is reserved word, therefore it can’t be used as a variable name. | Exit() can be used as a variable name. |
| No packages are included in order to use break statement in java | ****java.lang.System.exit()****  needs to be included in order to use exit(). |
| Syntaxbreak; | Syntaxsystem.exit(); |