**Break and continue and labeled break and labeled continue**

**Break**: The break statement is used to quit the loop without executing any of the remaining statements;

Exiting the loop entirely.

**An example of a program which contains a loop with jump statement as break: (statements like jump and continue as jump statements)**

//This is to check whether a no. is prime or not

import java.io.\*;

class Prime

{

public static void main(String args[])throws IOException

{

BufferedReader z=new BufferedReader(new InputStreamReader(System.in));

int no,flag=0;

System.out.print("Enter a no.:");

no=Integer.parseInt(z.readLine());

for(int i=2;i<=Math.sqrt(no);i++)

{

if(no%i==0)

{

flag=1;

break;

}

//otherwise the execution of the loop will be continued

}

if(flag==1)

{

System.out.println("It is not a prime no.");

}

else

{

System.out.println("It is a prime no.");

}

}

}

**A little understanding of the program:**

Here, to determine a proper factor of a no. (i.e. except 1 and the no itself) we run a loop from 2 to square root of the given no. Now, if we find a factor of the number in this range, then the number is clearly not a prime no. So, our motive is to find at least one factor of the given no in this range. Now, finding one factor in this range ensures us that the no. is not a prime factor. So, no further execution of the loop is needed. So, we use break statement such that when program finds one factor of the no. in the mentioned range the remaining part of the loop will not execute and the control will come out of the loop (to ensure no further loop is executed based on the given range. This makes program efficient and makes program execution time efficient)

**Continue:** The continue statement stops execution of the current iteration (i.e. skips the remaining part of the current execution) and goes back to the beginning of the loop to begin the next iteration.

There are situations when we need this type of thing. Then continue statement will become helpful.

Note: Both the break and continue cannot be used directly within a loop, they must appear within conditional checking statement blocks (if block, else block) in the loop.

**Labeled Break and continue**

Labeled break and continue are very much needed in nested loops (nested loop: when one loop takes place within another loop). Otherwise, the code becomes clumsy.(i.e. we have problems if we want the control to come out from both loops (in case of nested loop) , or we want to start execution for the very beginning of the first loop depending upon a mentioned condition in second loop)

Loop1: for (int i=0;i<10;i++)

{

Loop2: while(x<100)

{

y=i\*x;

if(y>500)

break Loop1;

…………………..

…………………..

}

}

Note: Loop1: for(int i=0;i<10;i++)

Here, we associate a name Loop1 with the loop. This is called labeling the loop.

And these labels become useful when we have to choose which loop to break or which loop to continue.