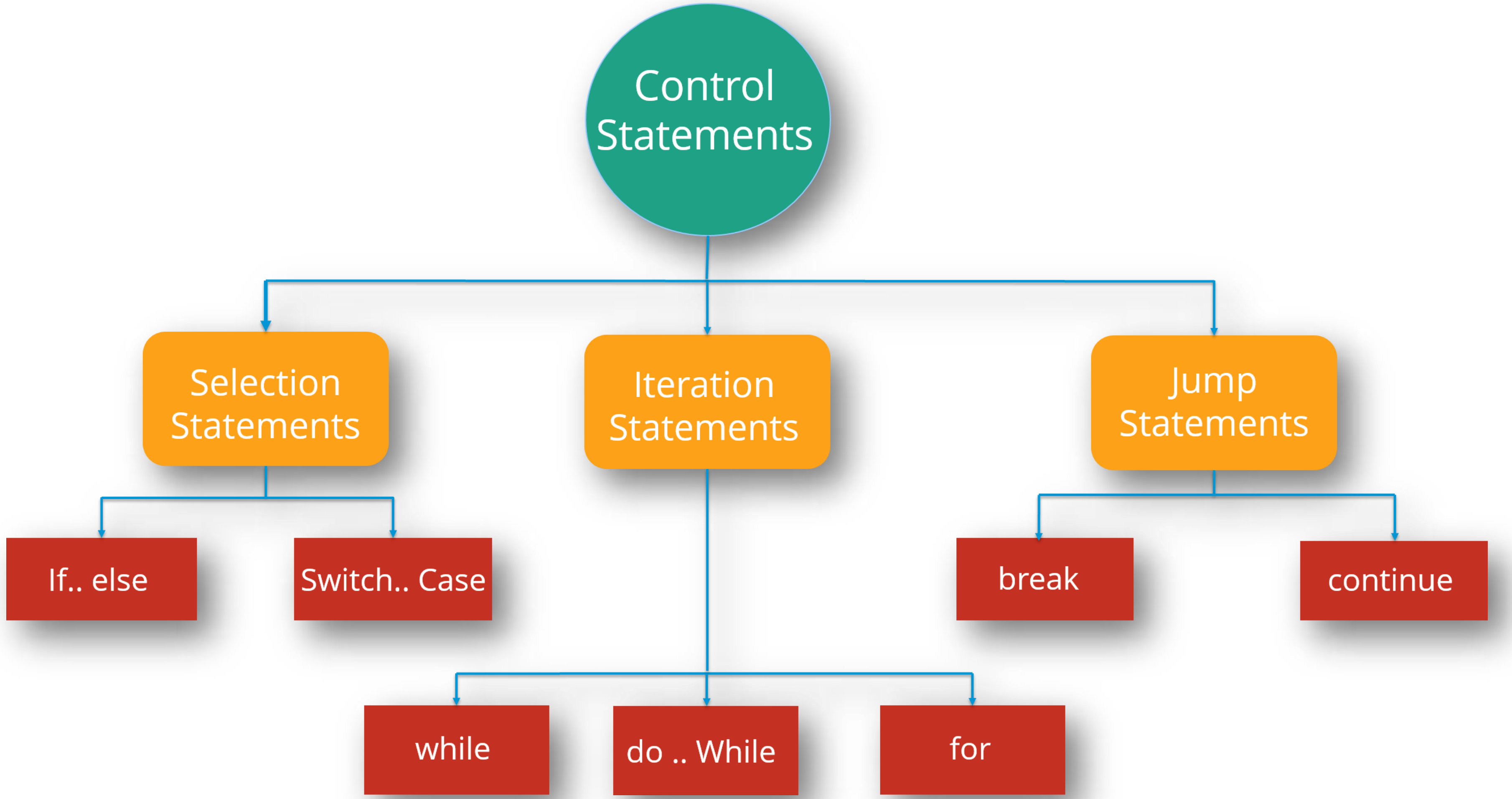
Java

Agenda

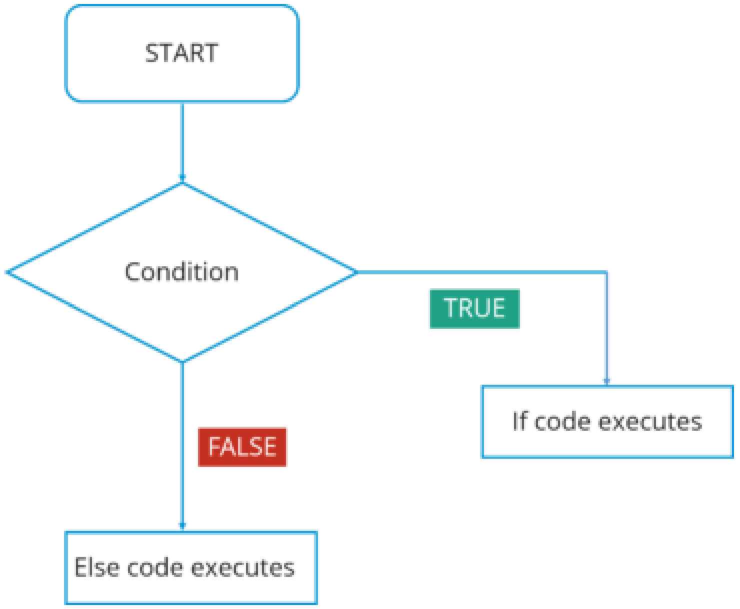
- Control Statements
- Conditional/Statements(Selection Statements)
- Loops /Iterative Statements
- Jump Statements

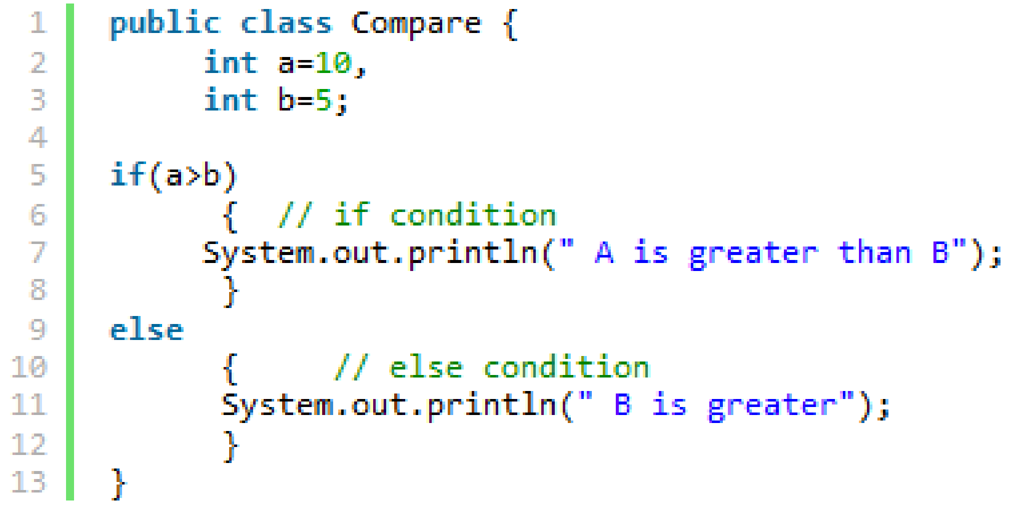
Control Statements



If..else

- In this flowchart, the code will respond in the following way:
  1. First of all, it will enter the loop where it checks the condition.
  2. If the condition is true, the set of statements in ‘if’ part will be executed.
  3. If the condition is false, the set of statements in the ‘else’ part will be executed.





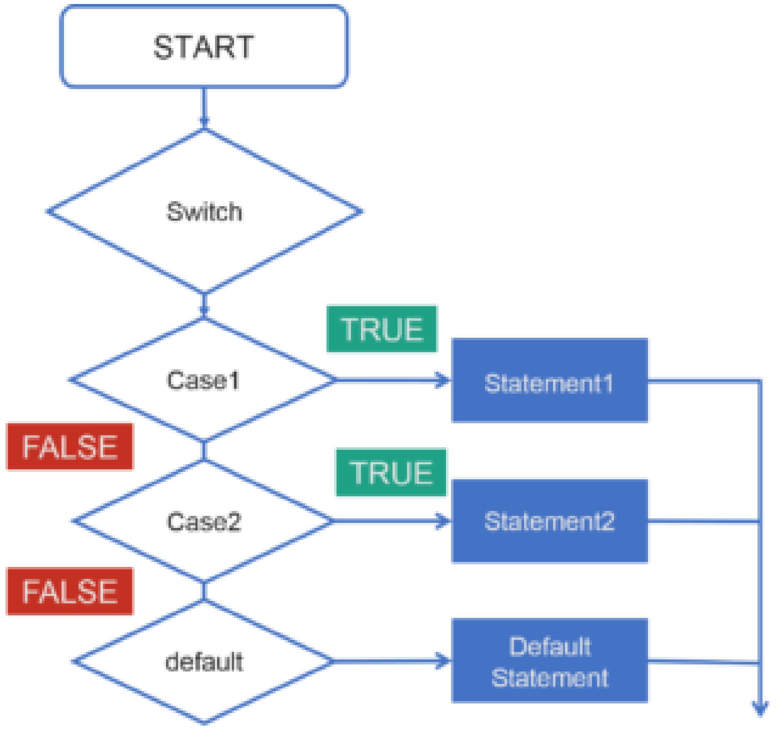
```
public class Compare {
      int a=10,
      int b=5;

if(a>b)
       {  // if condition
      System.out.println(" A is greater than B");
       }
else
       {     // else condition
       System.out.println(" B is greater");
       }
}
```

Switch..case

- The switch statement defines multiple paths for execution of a set of statements. It is a better alternative than using a large set of if-else statements as it is a multi-way branch statement.

- In this Switch case flowchart, the code will respond in the following steps:
  1. First of all it will enter the switch case which has an expression.
  2. Next it will go to Case 1 condition, checks the value passed to the condition. If it is true, Statement block will execute. After that, it will break from that switch case.
  3. In case it is false, then it will switch to the next case. If Case 2 condition is true, it will execute the statement and break from that case, else it will again jump to the next case.
  4. Now let's say you have not specified any case or there is some wrong input from the user, then it will go to the default case where it will print your default statement.

Switch..case (Cont..)

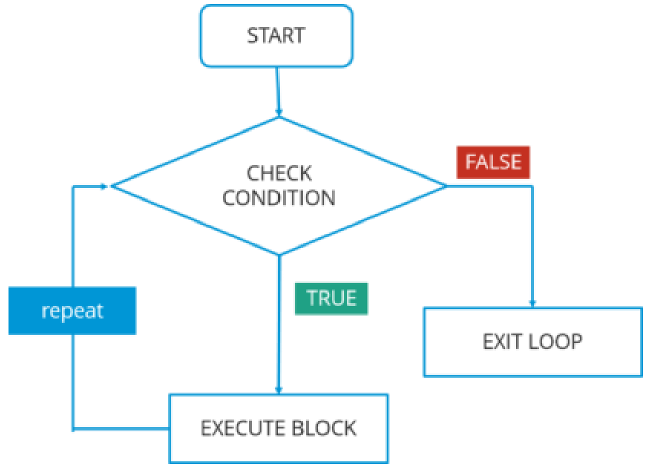


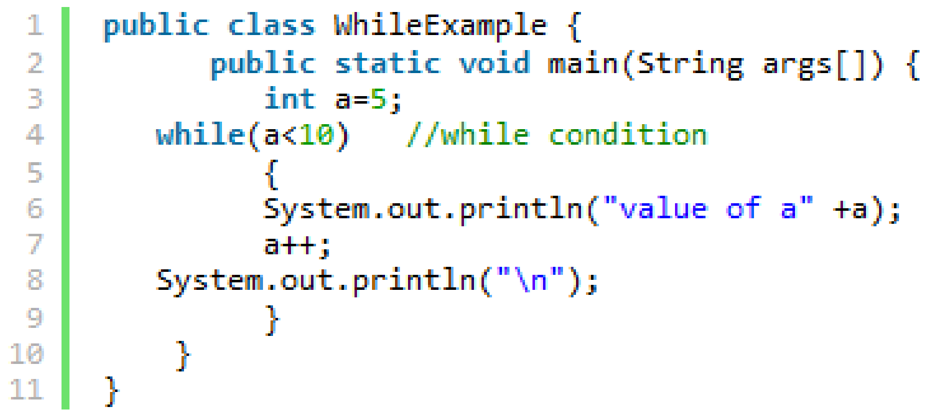
```
public class SwitchExample {
        int week=7;
        String weeknumber;

switch(week){     // switch case
case 1:
                weeknumber="Monday";
           break;

case2:
                weeknumber="tuesday";
           break;

case3:
                weeknumber="wednesday";
           break;

default:          // default case
                weeknumber="invalid week";
           break;
        }
   System.out.println(weeknumber);
        }
}
```

While loop

- **While statement:** Repeat a group of statements while a given condition is true. It tests the condition before executing the loop body.

- In this flowchart, the code will respond in the following steps:
  1. First of all, it will enter the loop where it checks the condition.
  2. If it's true, it will execute the set of code and repeat the process.
  3. If it's False, it will directly exit the loop.



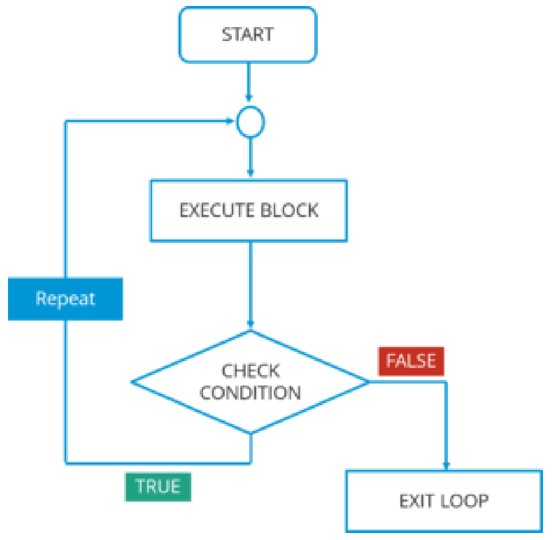


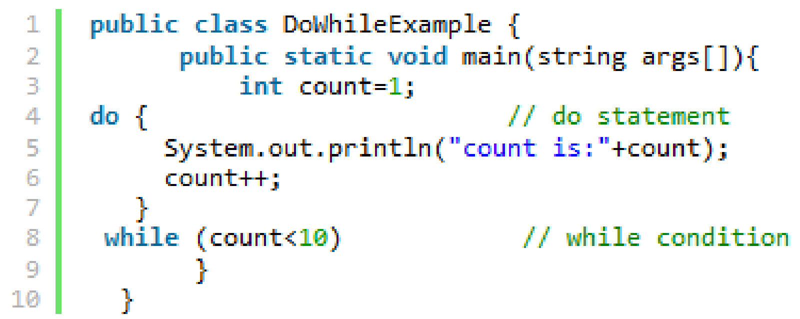
```
public class WhileExample {
        public static void main(String args[]) {
            int a=5;
    while(a<10)   //while condition
            {
            System.out.println("value of a" +a);
            a++;
    System.out.println("\n");
            }
     }
}
```

Do..while loop

- **Do-while statement:** It is like a while statement, but it tests the condition at the end of the loop body. Also, it will executes the program at least once.

- In this do-while flowchart, the code will respond in the following steps:
  1. First of all, it will execute a set of statements that is mentioned in your 'do' block.
  2. After that, it will come to 'while' part where it checks the condition.
  3. If the condition is true, it will go back and execute the statements.
  4. If the condition is false, it will directly exit the loop.





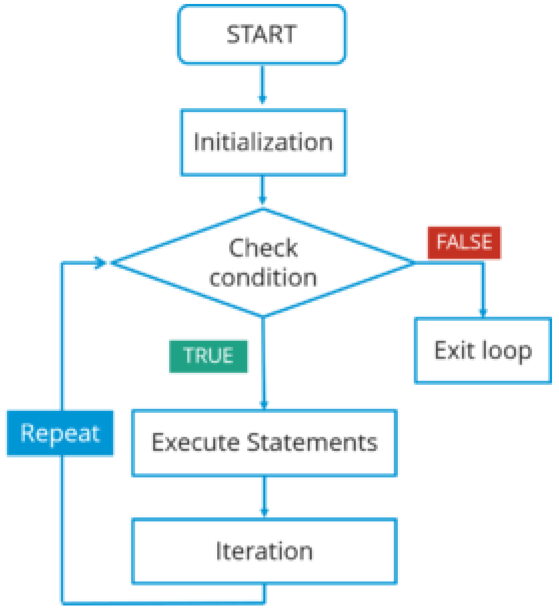
```
public class DoWhileExample {
      public static void main(string args[]){
          int count=1;
do {                         // do statement
     System.out.println("count is:"+count);
     count++;
   }
 while (count<10)             // while condition
          }
  }
```

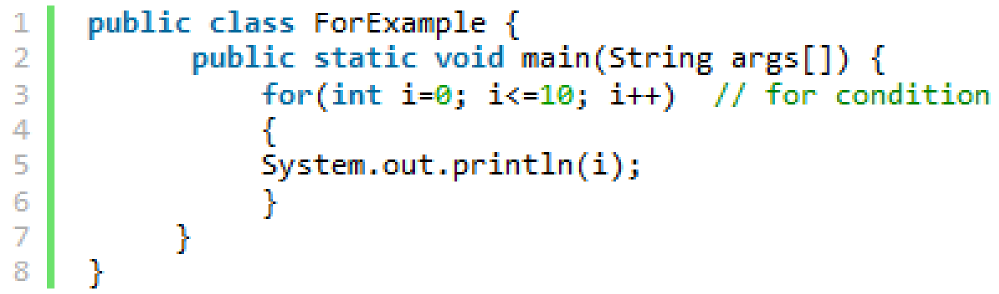
For loop

- **For statement:** For statement execute a sequence of statements multiple time where you can manage the loop variable. You basically have 3 operations here: initialization, condition and iteration.

- In this flowchart, the code will respond in the following steps:

  1. First of all, it will enter the loop where it checks the condition.
  2. Next, if the condition is true, the statements will be executed.
  3. If the condition is false, it directly exits the loop.





```
public class ForExample {
       public static void main(String args[]) {
            for(int i=0; i<=10; i++)  // for condition
            {
            System.out.println(i);
            }
       }
}
```

When we use while, do..while & for loops

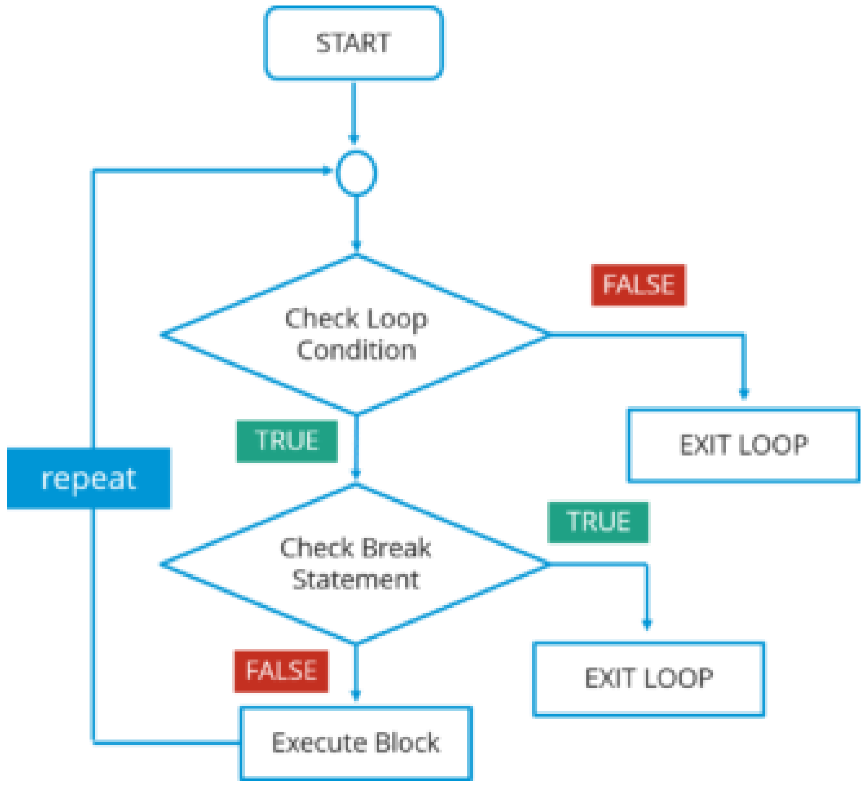
- **Scenario 1**: If u want to travel by your own vehicle(two wheeler or four wheeler),You should know how much petrol available in your vehicle and you know how much distance to travel.
- For loop --- Know the initial condition and no of iterations.
- **Scenario 2:** If u want to travel in a FLIGHT, You should buy a ticket then only you are eligible to enter into the Flight.
- While loop ---- First satisfy the condition then Proceed.
- **Scenario 3:** If u want to travel in BUS, You can board the bus then buy the ticket.
- Do..while ---- Proceed first then checking.

Jump Statements

- **Jump statement:** Jump statement are used to transfer the control to another part of your program. These are further classified into – *break* and *continue*.

Break statement

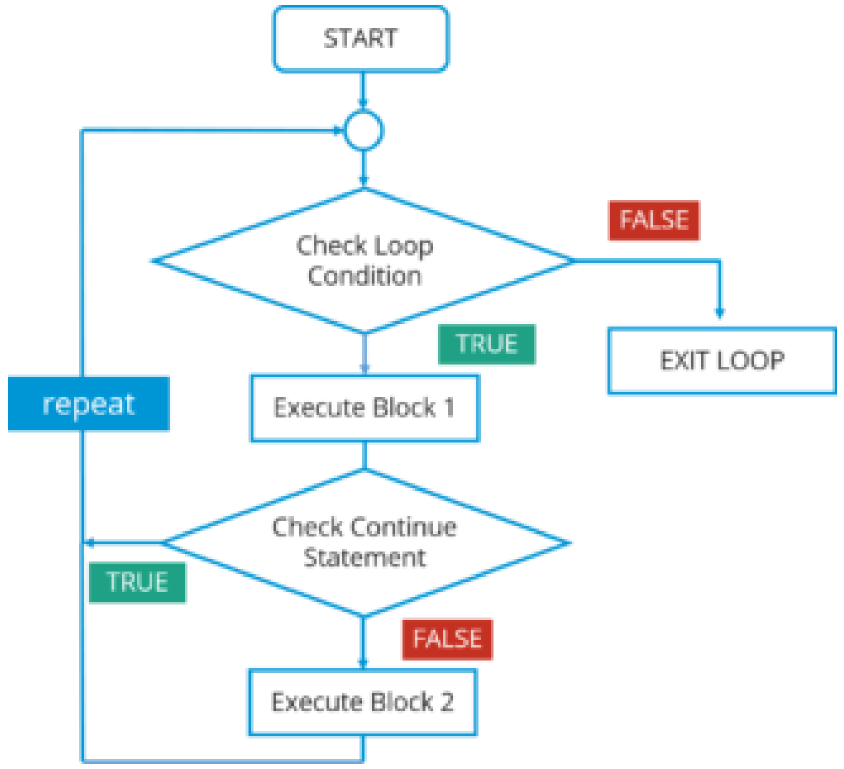
- **Break statement:** Whenever a break statement is used, the loop is terminated and the program control is resumed to the next statement following the loop
- In this flowchart, the code will respond in the following steps:
  1. First of all, it will enter the loop where it checks the condition.
  2. If the loop condition is false, it directly exits the loop.
  3. If the condition is true, it will then check the break condition.
  4. If break condition is true, it exists from the loop.
  5. If the break condition is false, then it will execute the statements that are remaining in the loop and then repeat the same steps.



Continue statement

- **Continue statement:** Continue statement is another type of control statements. The continue keyword causes the loop to immediately jump to the next iteration of the loop.

- In this flowchart, the code will respond in the following steps:
  1. First of all, it will enter the loop where it checks the condition. If the loop condition is false, it directly exits the loop.
  2. If the loop condition is true, it will execute block 1 statements.
  3. After that it will check for 'continue' statement. If it is present, then the statements after that will not be executed in the same iteration of the loop.
  4. If 'continue' statement is not present, then all the statements after that will be executed.



Input From the User

- Scanner class is to take the input from user which is imported from java.util.Scanner package.
  Scanner input = new Scanner(System.in);
- Get Integer Input From the User
  int number = input.nextInt();
- Get float, double and String Input
  float myFloat = input.nextFloat();
- double myDouble = input.nextDouble();
  String myString = input.next();

Assignment

1. Write a Java program to get a number from the user and print whether it is positive or negative
2. Write a Java program to find greatest of 3 numbers.
3. Write a Java to display the multiplication table of a given integer using for loop.
4. Write a Java program count the number of digits of the number using while loop.
5. Write a Java program to reverse a number using while loop.
6. Write a Java program to check Number is Palindrome or not using while loop.