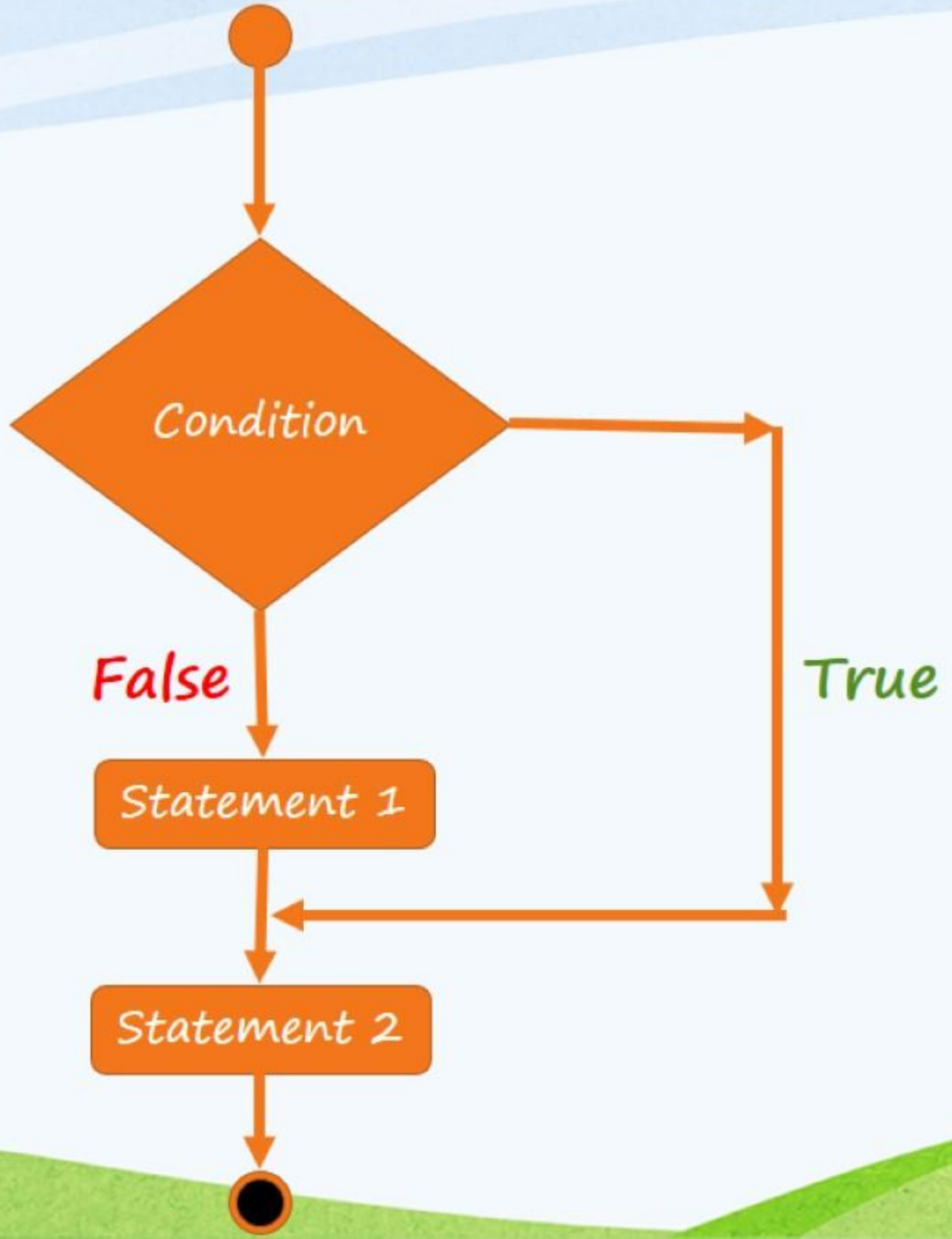
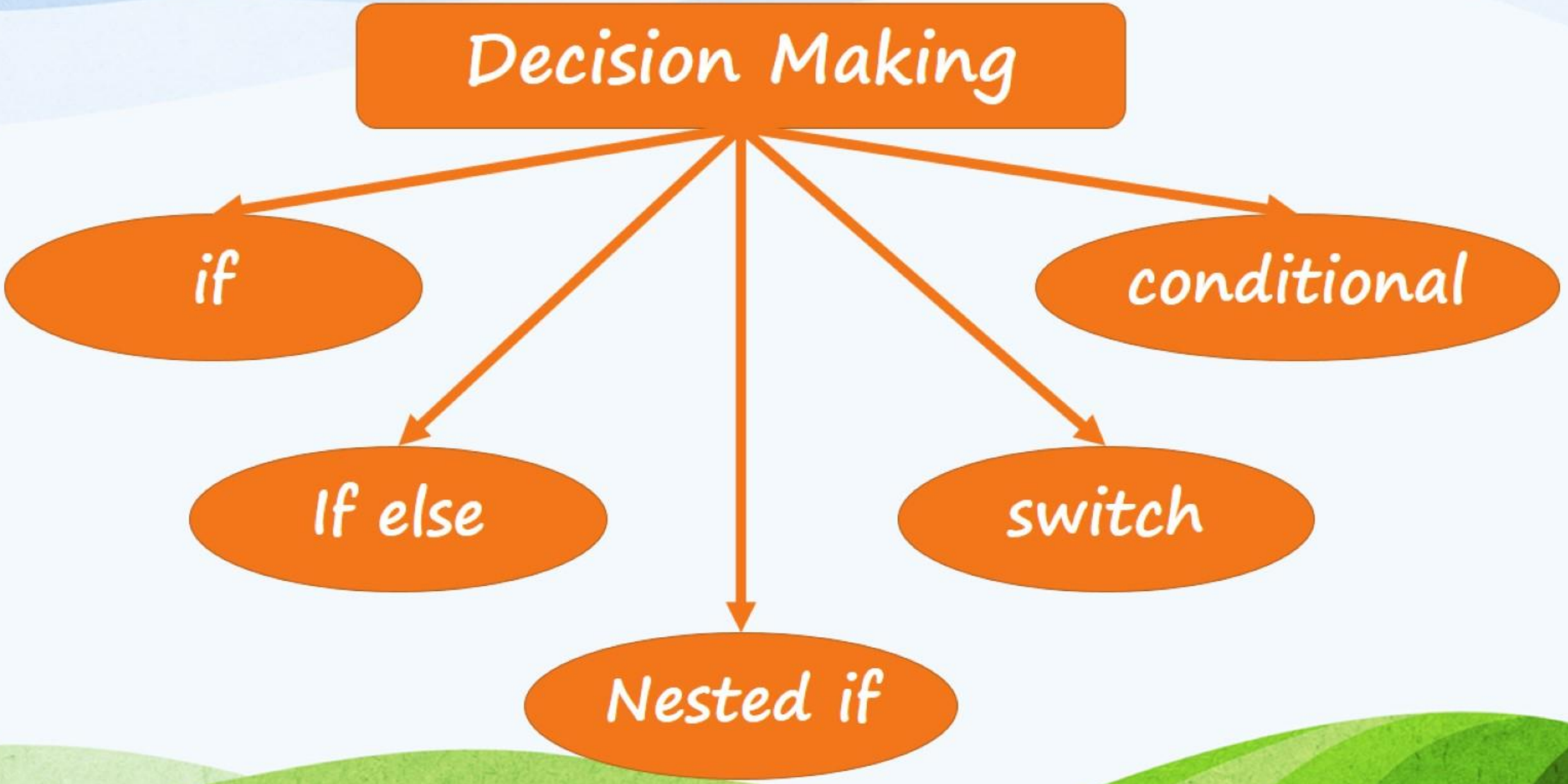

Decision Making

Decision Making

- Decision making structures have one or more conditions to be evaluated, along with a statement or statements that are to be executed if the condition is determined to be true, and optionally, other statements to be executed if the condition is determined to be false.

Flow Chart



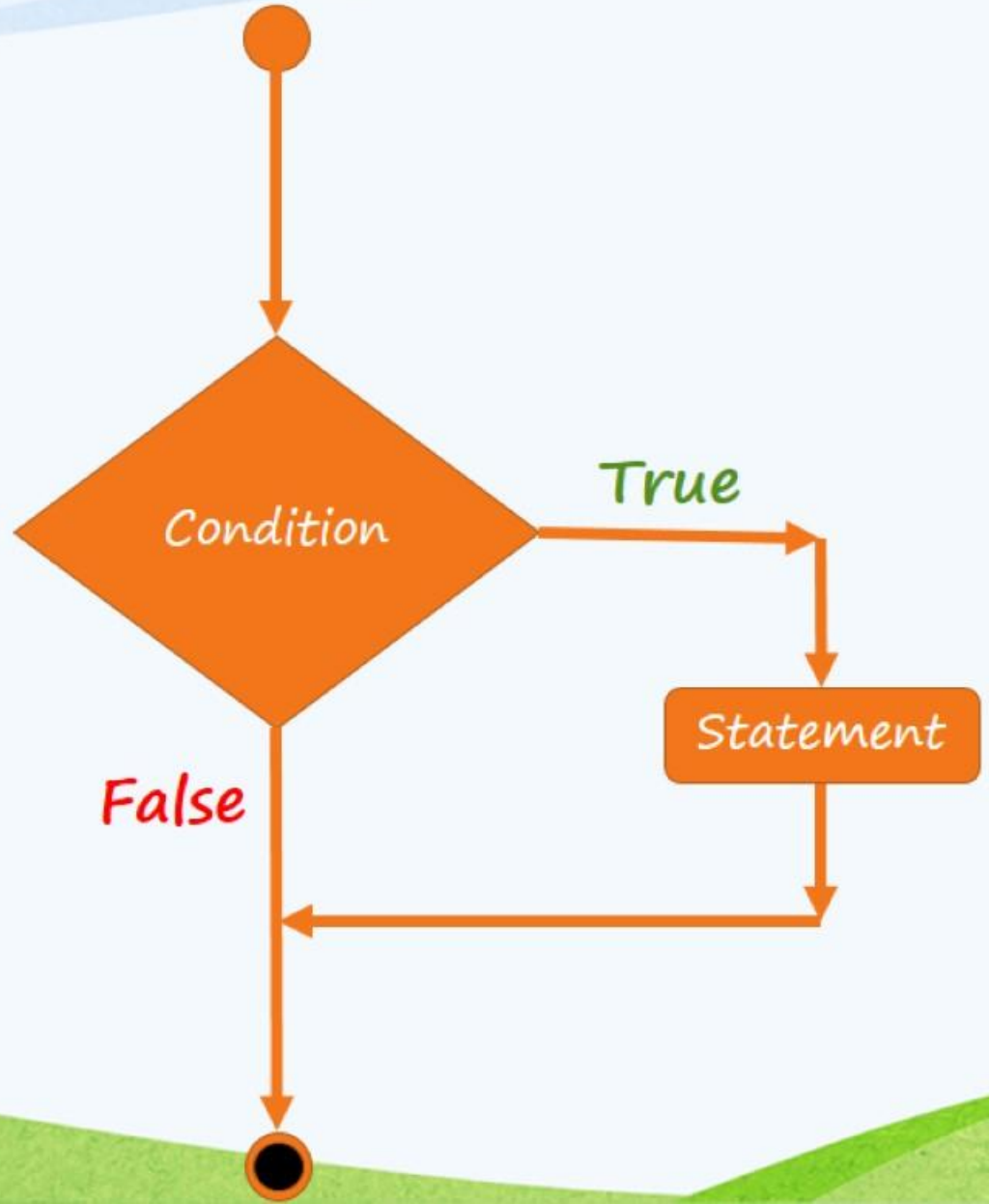

Decision Making
if
conditional
If else
switch
Nested if

if statement

if statement

- An if statement consists of a Condition followed by one or more statements.
- If the Condition evaluates to true then the block of code inside the if statement will be executed.
- If not, the first set of code after the end of the if statement will be executed.

Flow Chart



if statement Syntax

```
if ( condition )
  {
     //statement
  }
```

```
int age = 25;
if(age > 18)
{
System.out.println("Eligible");
}
```

Output: Eligible

```
String name = "Yash";
if(name == "Yash")
{
  System.out.println("Accepted");
}
```

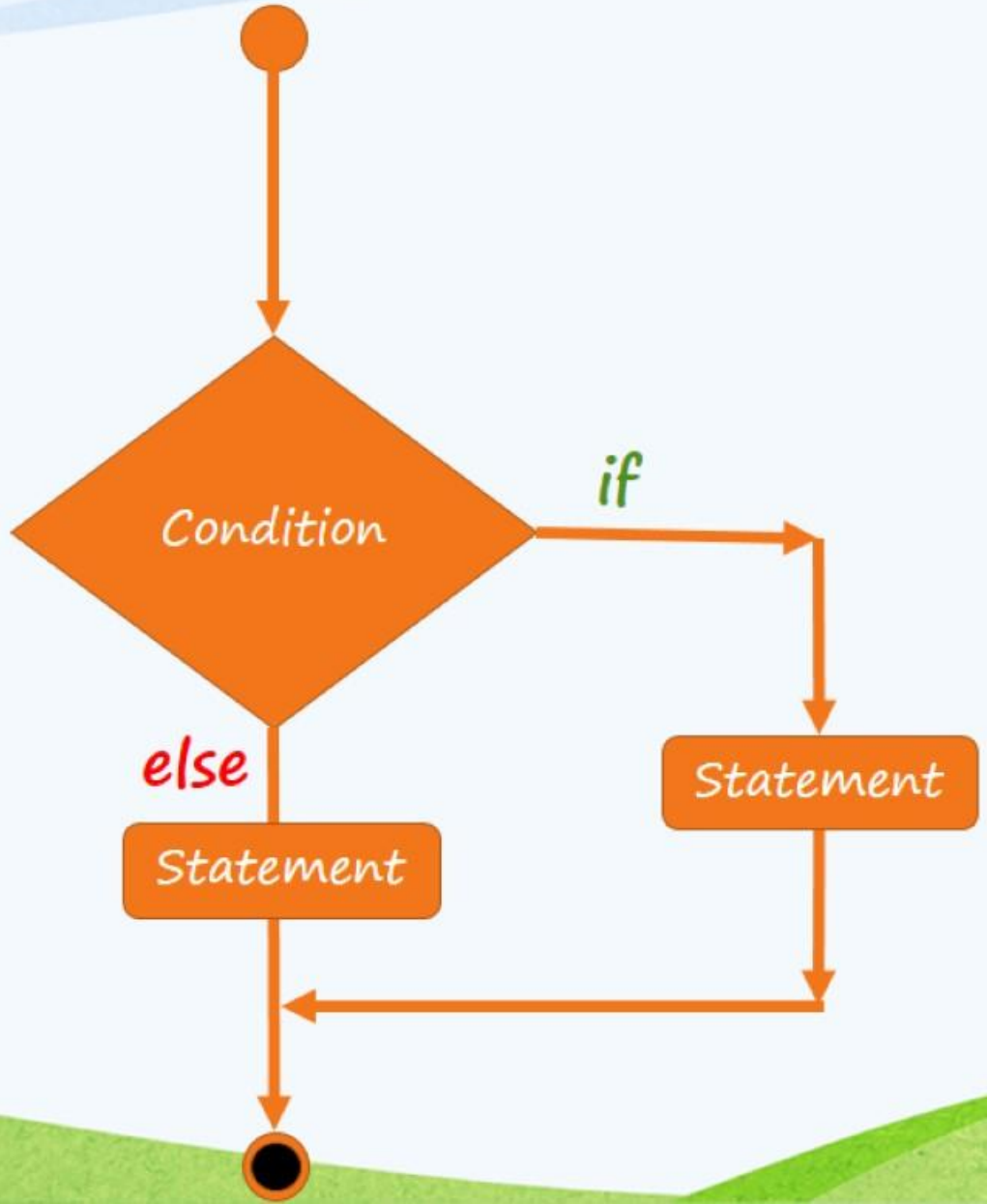
Output: Accepted

If else statement

If else statement

- The Java if-else statement also tests the condition.
- It executes the if block if condition is true otherwise else block is executed.

Flow Chart



if statement Syntax

```
if ( condition )
  {
     //statement
  }
else
{
    //statement
}
```

```
int age = 16;
if(age > 18)
{
 System.out.println("Eligible");
}
else {
System.out.println("Rejected");
}
```

Output: Rejected

```
String name = "Ash";
if(name == "Yash"){
System.out.println("Accepted");
        }
else{
System.out.println("Rejected");
        }
```

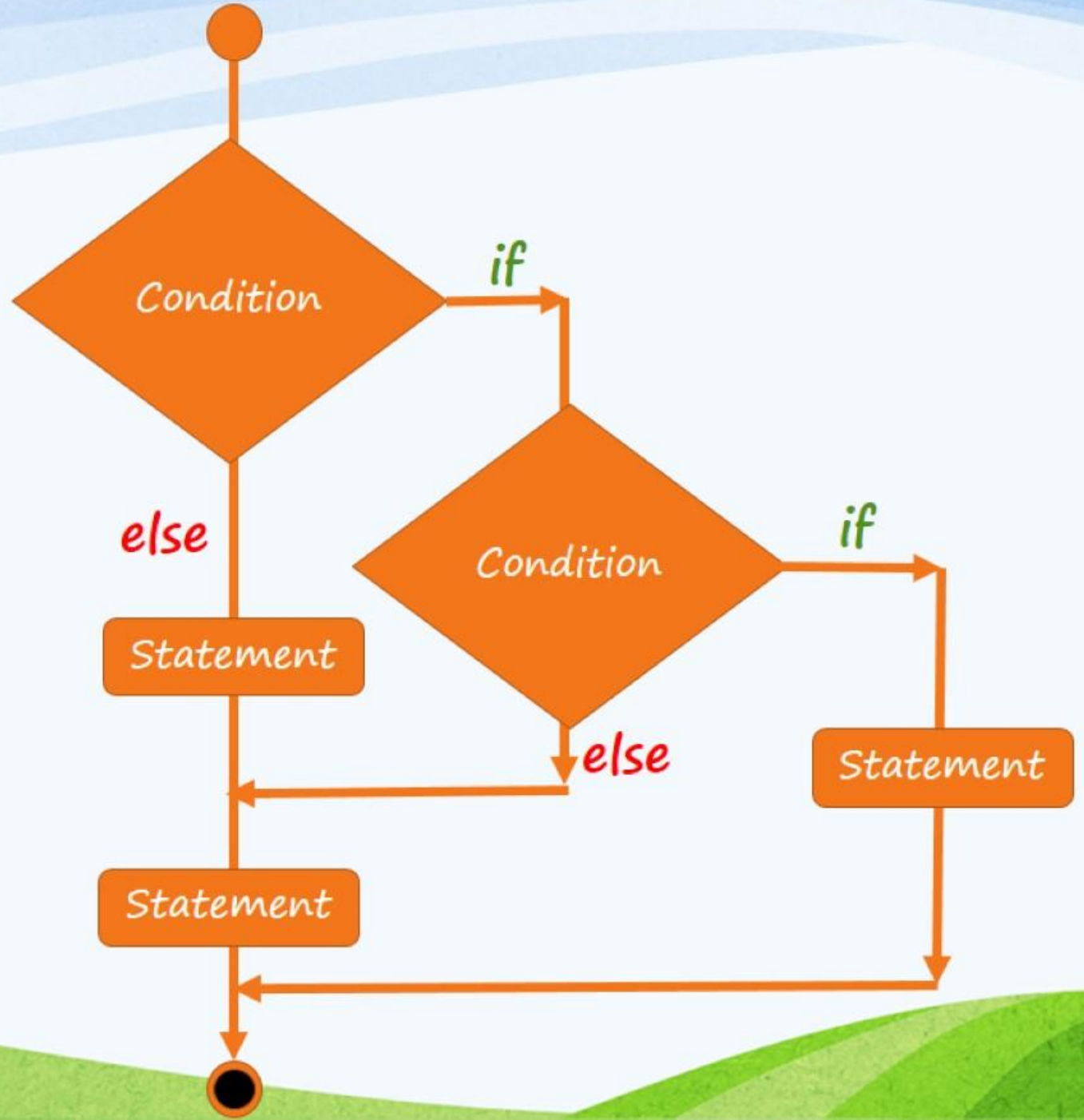
Output: Rejected

Nested If statement

Nested If statement

- You can use if or else if statements, inside another if or else if statement.

Flow Chart


Condition
if
else
Statement
Condition
if
else
Statement
Statement

Nested if statement Syntax

```
if ( condition1 )
  {
     if ( condition2 )
       {
          //statement
       }
  }
```

```
int age = 16;
if(age > 18)
  {
        if(age <60){

System.out.println("Eligible");
                        }
                }
```

Output: Eligible

```
String name = "Yash";
int age=25;
if(name == "Yash"){
  if(age > 18)
  {
   System.out.println("Accepted");
  }
}
```

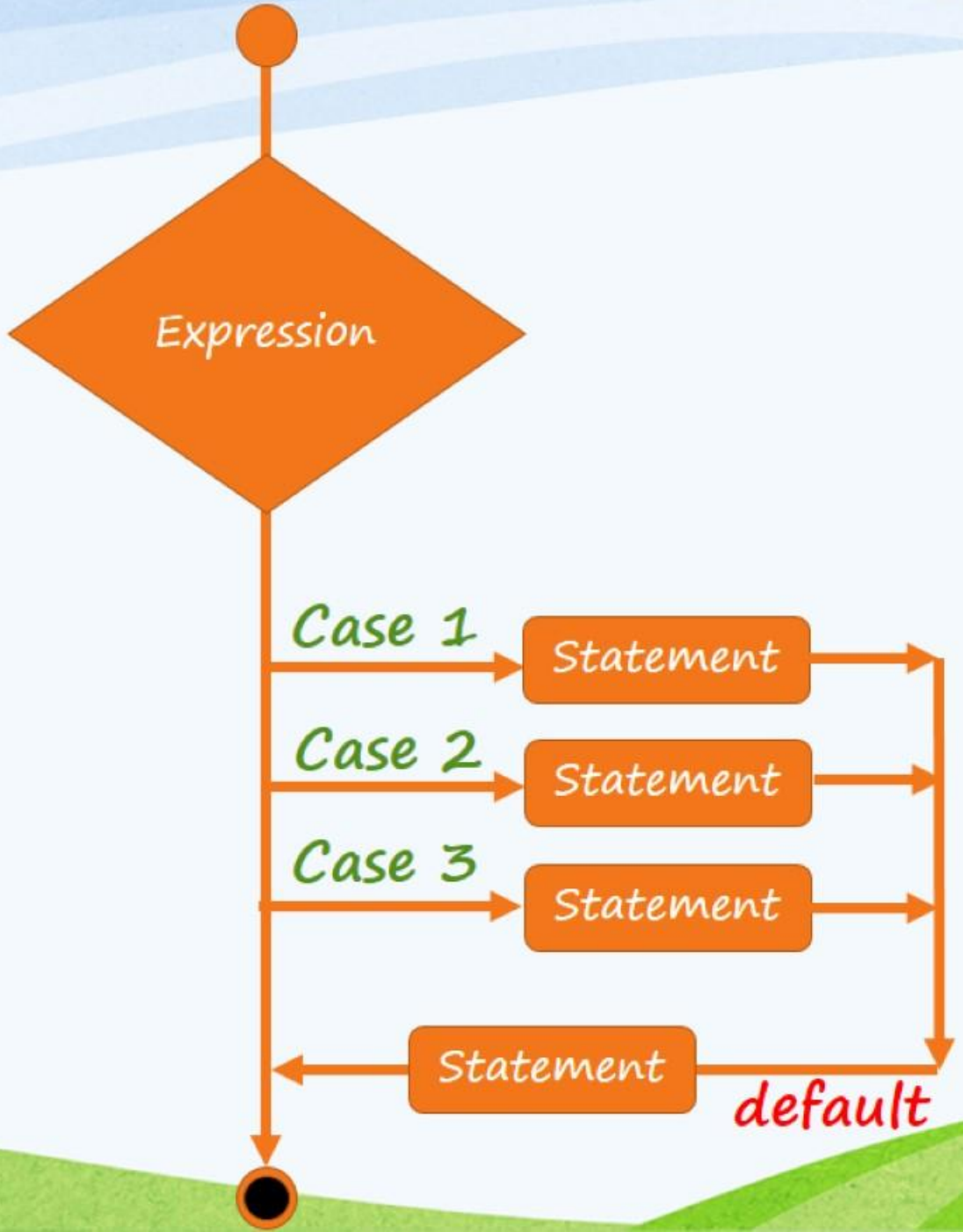
Output: Accepted

Switch statement

Switch statement

- A switch statement allows a variable to be tested for equality against a list of values.

Flow Chart


Expression
Case 1
Statement
Case 2
Statement
Case 3
Statement
Statement
default

Switch statement Syntax

```
switch(expression) {
                    case value :
                    // Statements
                    break; // optional

                    case value :
                    // Statements
                    break; // optional

                    // You can have n number of case statements.
                   default : // Optional
                   // Statements
                  }
```

```
int cgpa = 8;
        switch(cgpa) {
            case 7 :
                System.out.println("60-69");
                break;
            case 8 :
                System.out.println("70-79");
                break;
            case 9 :
                System.out.println("80-89");
            case 10 :
                System.out.println("90-100");
                break;
            default :
                System.out.println("Invalid cgpa");
        }
```

Output: 70-79

```
int cgpa = 12;
        switch(cgpa) {
            case 7 :
                System.out.println("60-69");
                break;
            case 8 :
                System.out.println("70-79");
                break;
            case 9 :
                System.out.println("80-89");
            case 10 :
                System.out.println("90-100");
                break;
            default :
                System.out.println("Invalid cgpa");
        }
```

Output: Invalid cgpa

Control statement

Loop control statements change execution from its normal sequence.

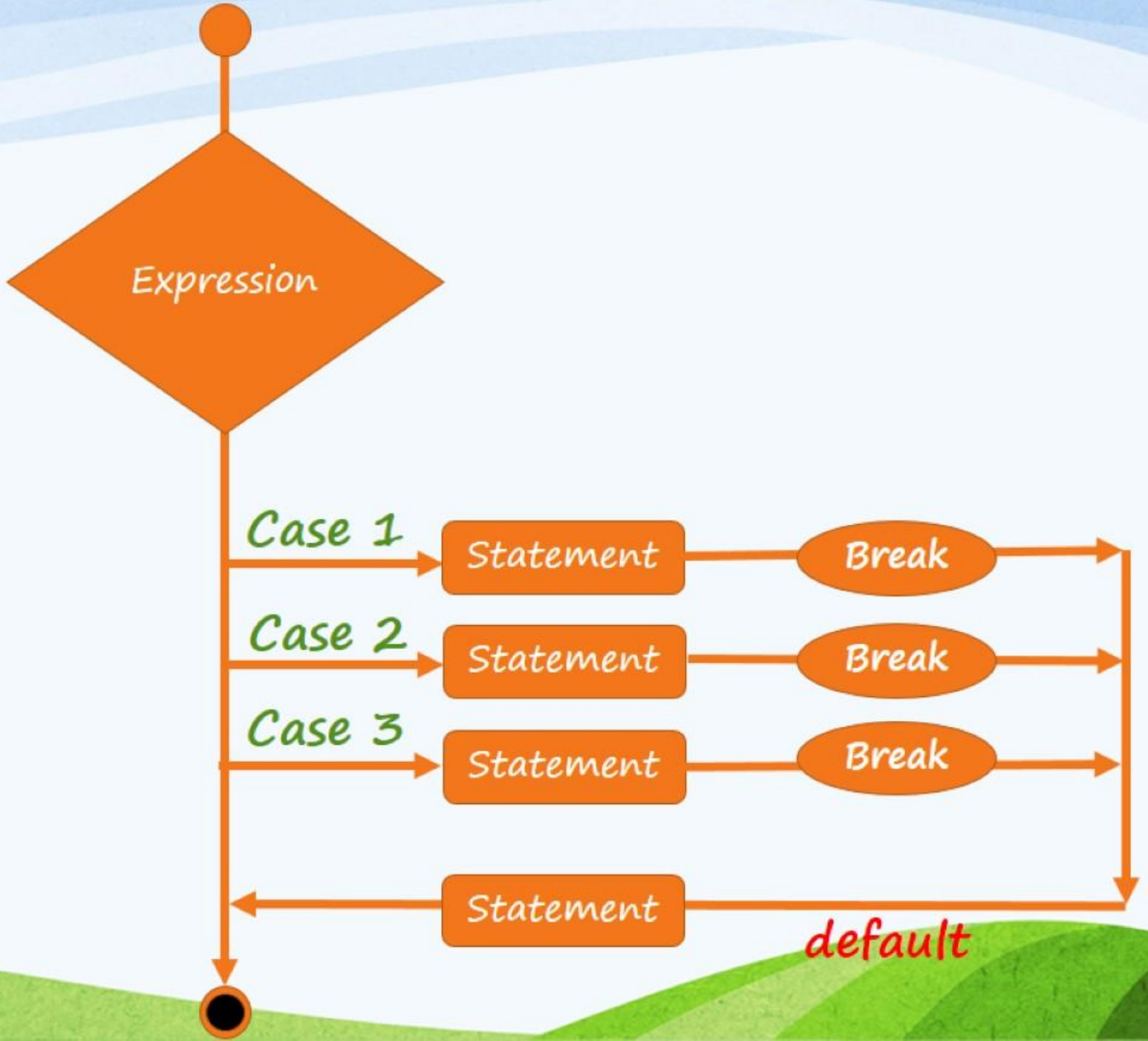
When execution leaves a scope, all automatic objects that were created in that scope are destroyed.

Break statement

Break statement

- Terminates the loop or switch statement and transfers execution to the statement immediately following the loop or switch.

Flow Chart


Expression
Case 1
Statement
Break
Case 2
Statement
Break
Case 3
Statement
Break
Statement
default

```
for(int i=0; i<5; i++)
  {
      if(i==3){
                  break;
               }
System.out.println(i);
  }
```

Output: 0

1

2

```
int cgpa = 12;
        switch(cgpa) {
            case 7 :
                System.out.println("60-69");
                break;
            case 8 :
                System.out.println("70-79");
                break;
            case 9 :
                System.out.println("80-89");
            case 10 :
                System.out.println("90-100");
                break;
            default :
                System.out.println("Invalid cgpa");
        }
```

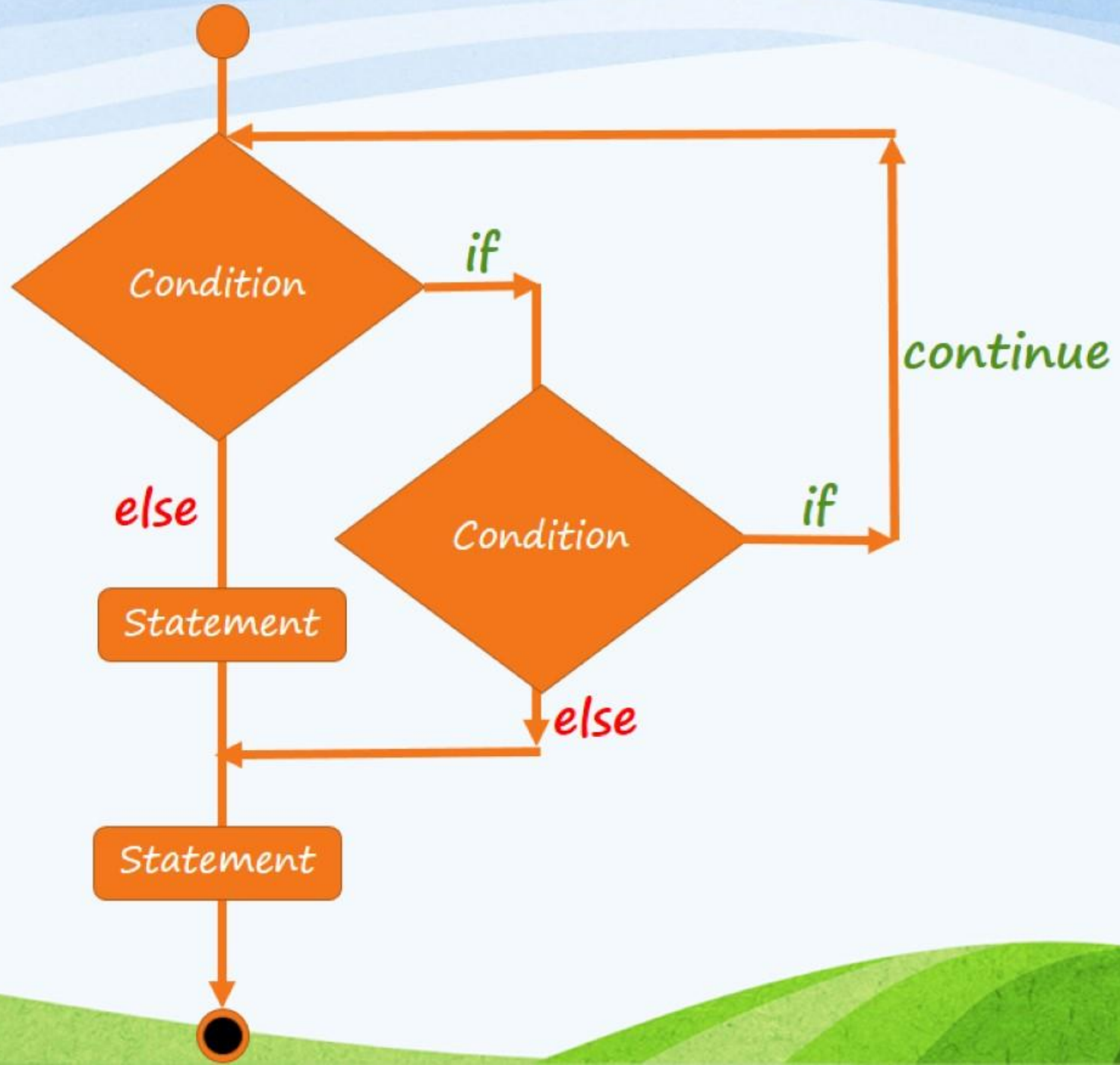
Output: Invalid cgpa

Continue statement

Continue statement

- The continue keyword can be used in any of the loop control structures.
- It causes the loop to immediately jump to the next iteration of the loop.

Flow Chart


Condition
if
continue
else
Condition
if
else
Statement
Statement

```
for(int i=0; i<5; i++)
  {
      if(i==3){
                  continue;
               }

System.out.println(i);
          }
```

Output: 0
1
2
4