NAME : Thejus Gowda PN
EMAIL: thejusgowdapn1059@gmail.com

Difference between if-else ladder and switch case

1. Expression Evaluation:
   - In an `if-else` ladder, each condition is evaluated sequentially from top to bottom until a true condition is found. Once a true condition is encountered, the corresponding block of code is executed, and the rest of the conditions are not evaluated.
   - In a `switch-case` statement, the expression (usually an integer or character) is evaluated once, and the program jumps directly to the matching `case` label. There is no sequential evaluation of conditions.

2. Supported Data Types:
   - `if-else` can handle complex conditions involving various data types, such as booleans, numbers, strings, and more. It allows for flexible and complex condition checking.
   - `switch-case` is usually used with integral types (like integers or characters) and can only compare the expression against constant values specified in the `case` labels. It doesn't support complex conditions.

3. Fall-Through Behavior:
   - In an `if-else` ladder, once a condition is met and its corresponding block is executed, the program exits the entire ladder. There's no inherent fall-through behavior.
   - In a `switch-case` statement, if a `case` label is executed, the program will continue executing subsequent `case` labels until a `break` statement is encountered or the `switch` block is exited. This allows for fall-through behavior, where multiple cases might be executed if not properly managed with `break` statements.

4. Use Cases:
   - `if-else` is more flexible and can handle complex conditions involving multiple variables and expressions. It is suitable when conditions are varied and may require different types of comparisons.
   - `switch-case` is useful when you have a single expression and you want to compare it against a set of constant values, each of which leads to a different action. It provides a clearer and more concise way to handle multiple cases with the same variable.

5. Readability and Maintainability:
   - An `if-else` ladder can become lengthy and harder to read as the number of conditions increases. Nested `if-else` ladders can be especially difficult to understand.
   - A well-organized `switch-case` statement can be more readable and maintainable when dealing with a larger number of cases. It's often easier to see the structure of the different cases and how they relate.

Nested Simple-if

It means an if statement inside another if satatement

Example:

```
Public class Demo{
        Public static void main(String [] args){
        int  x = 10;
        If(x>=5 && x<=100)
        {
                If(x>=5)
                {
                        System.out.println(“x is greater then 5”);
                }
        }

}

}
```

Nested If-else

It means an if- else statement inside and if else statement

```
public class NestedIfExample {
   public static void main(String[] args) {
      int x = 10;
      int y = 5;

      if (x > 0) {
         System.out.println("x is positive");

         if (y > 0) {
            System.out.println("y is positive");
         } else {
            System.out.println("y is non-positive");
         }
      } else {
         System.out.println("x is non-positive");
      }
   }
}
```