System Requirements Specification Index

For

If Statement

Version 1.0

IIHT Pvt. Ltd.
fullstack@iiht.com

Table of Contents

USE CASE DESCRIPTION
System Requirements Specification

1 Project Abstract

This project assesses knowledge of Java conditional statements, specifically the `if` statement. The tasks involve checking conditions and making decisions based on numerical values, logical comparisons, and data classifications.

2 Assessment Tasks

Task 1: Check if a Number is Positive or Negative:

- Declare an integer variable `number` with an initial value of `-5`.
- Use an `if` statement to check if `number` is greater than `0`.
  - ➔ If true, print `"The number <number> is positive."`.
- Use another `if` statement to check if `number` is less than or equal to `0`.
  - ➔ If true, print `"The number <number> is negative."`.

Task 2: Find the Largest of Three Numbers:

- Declare and initialize three integer variables:
  - ➔ `a` with the value `10`.
  - ➔ `b` with the value `20`.
  - ➔ `c` with the value `15`.
- Compare the values of `a`, `b`, and `c` using the following `if` statements:
  - ➔ Condition 1:
    - ➢ Use `if (a >= b && a >= c)` to check if `a` is greater than or equal to both `b` and `c`.
    - ➢ If true, print `"The largest number is: <a>"`.
  - ➔ Condition 2:
    - ➢ Use `if (b >= a && b >= c)` to check if `b` is greater than or equal to both `a` and `c`.
    - ➢ If true, print `"The largest number is: <b>"`.
  - ➔ Condition 3:
    - ➢ Use `if (c >= a && c >= b)` to check if `c` is greater than or equal to both `a` and `b`.
    - ➢ If true, print `"The largest number is: <c>"`.

Task 3: Check if a Number is Even or Odd:

- Declare an integer variable `num` with an initial value of `8`.

- Use an `if` statement to check if `num` is divisible by `2` using the condition `num % 2 == 0`.
  ➔ If true, print `"The number <num> is even."`
- Use another `if` statement to check if `num` is not divisible by `2` using the condition `num % 2 != 0`.
  ➔ If true, print `"The number <num> is odd."`.

Task 4: Check Eligibility to Vote:

- Declare an integer variable `age` with an initial value of `17`.
- Use an `if` statement to check if `age` is greater than or equal to `18` using the condition `age >= 18.`
  ➔ If true, print `"You are eligible to vote."`.
- Use another `if` statement to check if `age` is less than `18` using the condition `age < 18`.
  ➔ If true, print `"You are not eligible to vote."`.

Task 5: Calculate Grade Based on Marks:

- Declare an integer variable `marks` with an initial value of `85`.
- Use multiple `if` statements to determine the grade based on the following conditions:
- Condition 1:
  - ➢ Check if `marks` is greater than or equal to `90` using the condition `marks >= 90`.
  - ➢ If true, print `"Grade: A"`.
- Condition 2:
  - ➢ Check if `marks` is between `75` and `89` using the condition `marks >= 75 && marks < 90`.
  - ➢ If true, print `"Grade: B"`.
- Condition 3:
  - ➢ Check if `marks` is between `50` and `74` using the condition `marks >= 50 && marks < 75`.
  - ➢ If true, print `"Grade: C"`.
- Condition 4:
  - ➢ Check if `marks` is less than `50` using the condition `marks < 50`.
  - ➢ If true, print `"Grade: F"`.

Expected Output:

```
The number -5 is negative.
The largest number is: 20
The number 8 is even.
You are not eligible to vote.
Grade: B
```

3 Template Code Structure

3.1 Package: com.yaksha.assignment.IfStatementAssignment

Resources

Class/Interface	Description	Status
IfStatementAssignment(class)	• Main class demonstrating conditional checks using `if` statements. • Includes examples of: - Checking positive or negative numbers. - Finding the largest of three numbers. - Determining even or odd numbers. - Checking voting eligibility. - Calculating grades based on marks.	Need to be implemented.

4 Execution Steps to Follow

1. **All actions like build, compile, running application, running test cases will be through Command Terminal.**
2. **To open the command terminal the test takers, need to go to Application menu (Three horizontal lines at left top) Terminal New Terminal.**
3. **This editor Auto Saves the code.**

4. **If you want to exit(logout) and continue the coding later anytime (using Save & Exit option on Assessment Landing Page) then you need to use CTRL+Shift+B-command compulsorily on code IDE. This will push or save the updated contents in the internal git/repository. Else the code will not be available in the next login.**
5. **These are time bound assessments the timer would stop if you logout and while logging in back using the same credentials the timer would resume from the same time it was stopped from the previous logout.**
6. **To run your project use command:**
   mvn compile exec:java
   -Dexec.mainClass="com.yaksha.assignment.IfStatementAssignment"
7. **To test your project test cases, use the command**
   mvn test

8. **You need to use CTRL+Shift+B - command compulsorily on code IDE, before final submission as well. This will push or save the updated contents in the internal git/repository, and will be used to evaluate the code quality.**