**Learner Assignment Submission Format**

# Learner Details

* **Name:Rakshith G S** **Enrollment Number:**
* **Batch / Class:**

# ● Assignment: (Bridge Course Day 2)

● **Date of Submission:**



**Problem**

**Solving**

**Activity**

**2.1**

**1.**

**Program**

**Statement**

Age

Checker

**2.**

**Algorithm**

1.

Declare

an

integer

variable

myAge

and

assign

a

value

to

it.

2.

Use

comparison

operators

to

check

the

following

conditions:

-

myAge

is

equal

to

25.

-

myAge

is

greater

than

18.

-

myAge

is

less

than

or

equal

to

65.

-

myAge

is

not

equal

to

30.

3. Print the Boolean result of each expression.

# 3. Pseudocode

BEGIN

SET myAge = some value

PRINT "Is my age equal to 25? " + (myAge == 25)

PRINT "Is my age greater than 18? " + (myAge > 18)

PRINT "Is my age less than or equal to 65? " + (myAge <= 65)

PRINT "Is my age not equal to 30? " + (myAge != 30)

END

# 4. Program Code

|  |
| --- |
| public class D2\_1 { |
| public static void main(String[] args) { |
| int myAge = 30; |
|  |
| System  .  out  .  println  (  "Is    my    age    equal    to    25  ?    "    +    (  myAge    ==    25  ))  ;      System  .  out  .  println  (  "Is    my    age    greater    than    18  ?    "    +    (  myAge    >    18  ))  ; |
| System.out.println("Is my age less than or equal to 65? " + (myAge |
| <= 65)); |
| System.out.println("Is my age not equal to 30? " + (myAge != 30)); |
| } |
| } |
|  |

# 5. Test Cases

Present a table of test cases you used to validate your program. Include a mix of regular, boundary, and edge cases.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Test Case No.** | **Input** | **Expected Output** | **Actual Output** | **Status (Pass/Fail)** |
| **1** | 30 | false,true,true,fals e | false,true,true,f alse | pass |
| **2** | 25 | true,true,true,true | true,true,true,tr ue | pass |
| **3** | 17 | false,false,true,tru e | false,false,true, true | pass |



**6.**

**Screenshots**

**of**

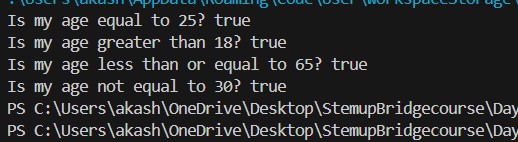
**Output**

**7.**

**Observation**

**/**

**Reflection**



This program demonstrates the use of comparison operators in Java to check different conditions. The comparison operators return a Boolean value (true or false) that can be printed directly.

**Problem Solving Activity 2.2**

# 1. Program Statement

Login Credentials

# 2. Algorithm

1. Declare the actual username and password.
2. Declare the entered username and password.
3. Use a logical AND operator (&&) to check if both the username and password match.
4. Store the result in a Boolean variable isValidLogin.



5.

Print

the

result.

**3.**

**Pseudocode**

BEGIN

SET

username

=

"akash"

SET

password

=

"akash123"

SET

enteredUsername

=

some

value

SET

enteredPassword

=

some

value

SET isValidLogin = (username == enteredUsername) AND (password == enteredPassword) PRINT "Is login valid? " + isValidLogin

END

|  |
| --- |
| **public class D2\_2 {** |
| **public static void main(String[] args) {** |
| **String username = "akash";** |
| **String password = "akash123";**  **String enteredUsername = "akash";** |
| **String enteredPassword = "akash123";** |
| **boolean isValidLogin = (username.equals(enteredUsername)) &&** |
| **(password.equals(enteredPassword));** |
| **System.out.println("Is login valid? " + isValidLogin);** |
| **}** |
| **}** |

# 4. Program Code



# 5. Test Cases

Present a table of test cases you used to validate your program. Include a mix of regular, boundary, and edge cases.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Test Case No.** | **Input** | **Expected Output** | **Actual Output** | **Status (Pass/Fail)** |
| **1** | same | true | true | pass |
| **2** | diff | false | false | pass |
|  |  |  |  |  |



**6.**

**Screenshots**

**of**

**Output**

**7.**

**Observation**

**/**

**Reflection**

This

program

demonstrates

the

use

of

logical

operators

in

Java

to

check

multiple

conditions.

The

&&

operator

returns

true

only

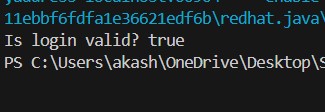
if

both

conditions

are

true.



**Problem Solving Activity 2.3**

# 1. Program Statement

Find Number Range

# 2. Algorithm

1. Declare an integer variable num and assign a value to it.
2. Use logical operators to check the following conditions:

- num is greater than 10 AND less than 20.



-

num

is

less

than

5

OR

greater

than

100.

3.

Store

the

results

in

Boolean

variables.

4.

Print

the

results.

**3.**

**Pseudocode**

BEGIN

SET

num

=

some

value

SET

isBetween10And20

=

(

num

>

10)

AND

(

num

<

20)

SET

isLessThan5OrGreaterThan100

=

num

(

<

5)

OR

(

num

>

100)

PRINT

"Is

num

between

10

and

?

20

"

+

isBetween10And20

PRINT "Is num less than 5 or greater than 100? " + isLessThan5OrGreaterThan100

END

# 4. Program Code

|  |
| --- |
| public class D3\_3 { |
| public static void main(String[] args) { |
| int num = 15; |
| boolean isBetween10And20 = (num > 10) && (num < 20); |
| boolean isLessThan5OrGreaterThan100 = (num < 5) || (num > 100); |
| System.out.println("Is num between 10 and 20? " + isBetween10And20); |
| System.out.println("Is num less than 5 or greater than 100? " + |
| isLessThan5OrGreaterThan100); } |
| } |



**5.**

**Test**

**Cases**

Present

a

table

of

test

cases

you

used

to

validate

your

program.

Include

a

mix

of

regular,

boundary,

and

edge

cases.

**Test**

**Case**

**No.**

**Input**

**Expected**

**Output**

**Actual**

**Output**

**Status**

**(**

**)**

**Pass/Fail**

**1**

15

true,false

true,false

pass

**2**

4

false,true

false,true

pass

**3**

65

false,false

false,false

pass

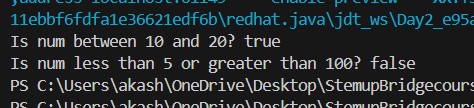
**6.**

**Screenshots**

**of**

**Output**

e.



# 7. Observation / Reflection

This program demonstrates the use of logical operators (&& and ||) in Java to check multiple conditions. The && operator returns true only if both conditions are true, while the || operator returns true if at least one condition is true.

**Problem Solving Activity 2.4**

# 1. Program Statement

Operator Precedence Challenge

Given the expression: 5+3\*2>10&&!(7==7)

# 4. Solution

5+3\*2>10&&!(7==7)



5+3\*2>10&&!(

true

)

5+3\*2>10

&&false

5+6>10

&&false

11>10

&&false

true&&false

false

**7.**

**Observation**

**/**

**Reflection**

This

challenge

demonstrates

the

importance

of

operator

precedence

in

programming.

The

order

in

which

operators

are

evaluated

can

significantly

affect

the

result

of

an

expression.

**Problem Solving Activity 2.5**

# 1. Program Statement

Check positive, negative or zero

# 2. Algorithm

1. Get an integer input from the user.
2. Use an if-else if-else structure to determine the sign of the number:

- If the number is greater than 0, print "Positive".



-

If

the

number

is

less

than

0

,

print

"Negative".

-

If

the

number

is

exactly

0

,

print

"Zero".

**3.**

**Pseudocode**

BEGIN

INPUT

num

IF

num

>

0

PRINT

"Positive"

ELSE

IF

num

<

0

PRINT

"Negative"

ELSE

PRINT "Zero"

END

# 4. Program Code

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| import java.util.Scanner; | | | | | | |
| public class D2\_5 { | | | | | | |
| public static void main(String[] args) { | | | | | | |
| **Test Case No.** | **Input** | **Expected Output** | **Actual Output** | **Status (Pass/Fail)** |
| **1** | -4 | negative | negative | pass |
| **2** | 5 | positive | positive | pass |
| **3** | 0 | zero | zero | pass |

|  |
| --- |
| Scanner scanner = new Scanner(System.in); |
| System.out.print("Enter an integer: "); |
| int num = scanner.nextInt(); |
| scanner.close(); if (num > 0) { |
| System.out.println("Positive"); |
| } else if (num < 0) { |
| System.out.println("Negative"); |
| } else { |
| System.out.println("Zero"); |
| }  } |
| } |
|  |



**5.**

**Test**

**Cases**

Present

a

table

of

test

cases

you

used

to

validate

your

program.

Include

a

mix

of

regular,

boundary,

and

edge

cases.

**6.**

**Screenshots**

**of**

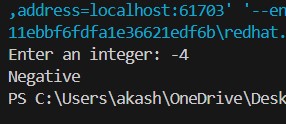
**Output**

**7.**

**Observation**

**/**

**Reflection**



This

program

demonstrates

a

simple

way

to

determine

the

sign

of

a

number

using

an

if-else

if-else

structure.

The

structure

checks

the

conditions

in

order

and

prints

the

corresponding

result.

**Problem Solving Activity 2.6**

# 1. Program Statement

Eligibility checking for driving

# 2. Algorithm

1. Get the user's age as input.
2. Use an if-else structure to determine if the user is eligible to drive:

- If the age is 18 or above, print "You are eligible to drive."



-

If

the

age

is

below

18

,

print

"You

are

not

eligible

to

drive.

You

need

to

be

at

least

18

years

old."

**3.**

**Pseudocode**

BEGIN

INPUT

age

IF

age

>=

18

PRINT

"You

are

eligible

to

drive."

ELSE

PRINT "You are not eligible to drive. You need to be at least 18 years old."

END

# 4. Program Code

|  |
| --- |
| import java.util.Scanner; |
| public class D2\_6 { |
| public static void main(String[] args) { |
| Scanner scanner = new Scanner(System.in); |

|  |
| --- |
| System.out.print("Enter your age: "); |
| int age = scanner.nextInt(); |
| scanner.close(); |
| if (age >= 18) { System.out.println("You are eligible to drive."); |
| } else { |
| System.out.println("You are not eligible to drive. You need to |
| be at least 18 years old."); |
| } |
| } |
| } |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Test Case No.** | **Input** | **Expected Output** | **Actual Output** | **Status (Pass/Fail)** |
| **1** | 19 | You are eligible to drive | You are eligible to drive | pass |
| **2** | 15 | You are not eligible to drive. | You are not eligible to drive. | pass |
| **3** | 18 | You are eligible to drive | You are eligible to drive | pass |



**5.**

**Test**

**Cases**

Present

a

table

of

test

cases

you

used

to

validate

your

program.

Include

a

mix

of

regular,

boundary,

and

edge

cases.

**6.**

**Screenshots**

**of**

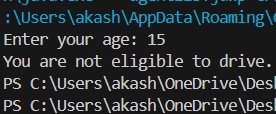
**Output**

**7.**

**Observation**

**/**

**Reflection**



This

program

demonstrates

a

simple

way

to

determine

eligibility

based

on

a

condition

(

age

>=

18)

using

an

if-else

structure.

The

structure

checks

the

condition

and

prints

the

corresponding

result.

**Problem Solving Activity 2.7**

# 1. Program Statement

Simple calculator using if-else if-else

# 2. Algorithm

1. Get two double numbers and an operator from the user.
2. Use if-else if-else to perform the operation based on the operator:

- If the operator is '+', add the numbers.



-

If

the

operator

is

'-',

subtract

the

numbers.

-

If

the

operator

is

'\*',

multiply

the

numbers.

-

If

the

operator

is

'/',

divide

the

numbers,

but

check

for

division

by

zero.

-

If

the

operator

is

none

of

the

above,

print

an

error

message.

**3.**

**Pseudocode**

BEGIN

INPUT

num1

INPUT

operator

INPUT

num2

IF operator == '+'

PRINT "Result: " + (num1 + num2)

ELSE IF operator == '-'

PRINT "Result: " + (num1 - num2)

ELSE IF operator == '\*'

PRINT "Result: " + (num1 \* num2)

ELSE IF operator == '/'

IF num2 != 0

PRINT "Result: " + (num1 / num2)

ELSE

PRINT "Error: Division by zero is not allowed."

ELSE

PRINT "Error: Invalid operator."

END

# 4. Program Code

|  |
| --- |
| import java.util.Scanner; |
| public class D2\_7 { |
|  |
| public    static    void    main  (  String  []    args  )    { |
| Scanner scanner = new Scanner(System.in);  System.out.print("Enter the first number: "); |
| double num1 = scanner.nextDouble(); |
| System.out.print("Enter the operator (+, -, \*, /): "); |
| char operator = scanner.next().charAt(0); |
| System.out.print("Enter the second number: "); |
| double num2 = scanner.nextDouble(); |
| scanner.close(); |
| if (operator == '+') { |
| System.out.println("Result: " + (num1 + num2)); |
| } else if (operator == '-') { |
| System.out.println("Result: " + (num1 - num2)); |

|  |
| --- |
| } else if (operator == '\*') { |
| System.out.println("Result: " + (num1 \* num2)); |
| } else if (operator == '/') { |
| if (num2 != 0) { System.out.println("Result: " + (num1 / num2)); |
| } else { |
| System.out.println("Error: Division by zero is not |
| allowed."); |
| } |
| } else { |
| System.out.println("Error: Invalid operator.");  } |
| } |
| } |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Test Case No.** | **Input** | **Expected Output** | **Actual Output** | **Status (Pass/Fail)** |
| **1** | 5,/,0 | Error: Division by zero is not allowed. | Error: Division by zero is not allowed. | pass |
| **2** | 5,\*,3 | 15 | 15 | pass |
| **3** | 6,-,3 | 3 | 3 | pass |



**5.**

**Test**

**Cases**

Present

a

table

of

test

cases

you

used

to

validate

your

program.

Include

a

mix

of

regular,

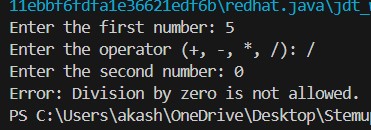
boundary,

and

edge

cases.

# 6. Screenshots of Output



# 7. Observation / Reflection



This

program

demonstrates

a

simple

way

to

implement

a

calculator

using

if-else

if-else

structure.

The

program

handles

division

by

zero

and

invalid

operators.

**Problem Solving Activity 2.8**

# 1. Program Statement

Movie ticket price based on their age.

# 2. Algorithm

1. Get the user's age and student status as input.
2. Use nested if or logical operators to determine the movie ticket price:

- If the user is under 5 or over 65, the price is $5.



-

If

the

user

is

between

5

and

18

)

(

inclusive

and

is

a

student,

the

price

is

$8.

-

Otherwise,

the

price

is

$12.

**3.**

**Pseudocode**

BEGIN

INPUT

age

INPUT

isStudent

IF

age

<

5

OR

age

>

65

price

=

5.0

ELSE IF age >= 5 AND age <= 18 AND isStudent

price = 8.0

ELSE

price = 12.0

PRINT "The movie ticket price is: $" + price

END

# 4. Program Code

|  |
| --- |
| import java.util.Scanner; |
| public class D2\_8 { |
| public static void main(String[] args) { |
| Scanner scanner = new Scanner(System.in);  System.out.print("Enter your age: "); |
| int age = scanner.nextInt(); |
| System.out.print("Are you a student? (true/false): "); |
| boolean isStudent = scanner.nextBoolean(); |
| scanner.close(); |
| double price; |
| if    (  age    <    5    ||    age    >    65  )    {      price    =    5.0  ; |
| } else if (age >= 5 && age <= 18 && isStudent) { |
| price = 8.0; |
| } else { |
| price = 12.0; |
| } |
| System.out.println("The movie ticket price is: $" + price);  } |
| } |

# 5. Test Cases

Present a table of test cases you used to validate your program. Include a mix of regular, boundary, and edge cases.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Test Case No.** | **Input** | **Expected**  **Output** | **Actual Output** | **Status (Pass/Fail)** |
| **1** | 4,true | $5 | $5 | pass |
| **2** | 66,false | $5 | $5 | pass |
| **3** | 35,false | $12 | $12 | pass |



**6.**

**Screenshots**

**of**

**Output**

**7.**

**Observation**

**/**

**Reflection**

This

program

demonstrates

a

simple

way

to

determine

the

movie

ticket

price

based

on

the

user's

age

and

student

status.

The

program

uses

logical

operators

to

combine

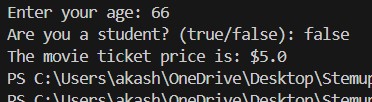
conditions

and

determine

the

price.



**Problem Solving Activity 2.9**

# 1. Program Statement

Find the day of the week depending on number(1-7).

# 2. Algorithm

1. Get the user's input as an integer.
2. Use a switch statement to determine the corresponding day of the week:

- Each case corresponds to a specific day (1 = Sunday, 3 = Tuesday, etc.).



-

The

default

case

handles

invalid

inputs.

**3.**

**Pseudocode**

BEGIN

INPUT

day

SWITCH

day

CASE

1:

PRINT

“Sunday”

CASE

2:

PRINT

“Monday”

CASE 3: PRINT “Tuesday”

CASE 4: PRINT “Wednesday”

CASE 5: PRINT “Thursday”

CASE 6: PRINT “Friday”

CASE 7: PRINT “Saturday”

DEFAULT: PRINT “Invalid input. Please enter a number between 1 and 7.”

END

# 4. Program Code

|  |
| --- |
| import java.util.Scanner; |
| public class D2\_9 { |
| public static void main(String[] args) { |
| Scanner scanner = new Scanner(System.in);  System.out.print("Enter a number (1-7): "); |
| int day = scanner.nextInt(); |
| scanner.close(); |
| switch (day) { |
| case    1  : |
| System.out.println("Sunday"); |
| break; case 2: |
| System.out.println("Monday"); |
| break; |
| case 3: |
| System.out.println("Tuesday"); |
| break; |
| case 4: System.out.println("Wednesday"); |
| break; |
| case 5: |
| System.out.println("Thursday"); |

|  |
| --- |
| break; |
| case 6: |
| System.out.println("Friday"); |
| break; case 7: |
| System.out.println("Saturday"); |
| break; |
| default: |
| System.out.println("Invalid input. Please enter a number |
| between 1 and 7."); |
| }  } |
| } |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Test Case No.** | **Input** | **Expected Output** | **Actual Output** | **Status (Pass/Fail)** |
| **1** | 1 | sunday | sunday | pass |
| **2** | 5 | Thursday | Thursday | pass |
| **3** | 8 | Invalid input. Please enter a number between 1 and 7. | Invalid input. Please enter a number between 1 and 7. | pass |



**5.**

**Test**

**Cases**

Present

a

table

of

test

cases

you

used

to

validate

your

program.

Include

a

mix

of

regular,

boundary,

and

edge

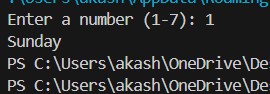
cases.

**6.**

**Screenshots**

**of**

**Output**



# 7. Observation / Reflection

This program demonstrates a simple way to use a switch statement to determine the day of the week based on the user's input. The switch statement makes the code more readable and **Problem Solving Activity 2.10**



efficient.

# 1. Program Statement

Simple Menu Selection,Simulate an ATM.

# 2. Algorithm

1. Display the ATM menu to the user.
2. Get the user's input as an integer.
3. Use a switch statement to determine the action based on the user's input:



-

Each

case

corresponds

to

a

specific

action

(1

=

Check

Balance,

2

=

Withdraw,

etc.).

-

The

default

case

handles

invalid

inputs.

**3.**

**Pseudocode**

BEGIN

DISPLAY

"ATM

Menu:"

DISPLAY

"1.

Check

Balance"

DISPLAY

"2.

Withdraw"

DISPLAY

"3.

Deposit"

DISPLAY

"4.

Exit"

INPUT choice

SWITCH choice

CASE 1: PRINT "You have chosen to check your balance."

CASE 2: PRINT "You have chosen to withdraw money."

CASE 3: PRINT "You have chosen to deposit money."

CASE 4: PRINT "Exiting..."

DEFAULT: PRINT "Invalid choice. Please try again."

END

# 4. Program Code

|  |
| --- |
| import java.util.Scanner; |
| public class D2\_10 { |
| public static void main(String[] args) { |
| Scanner scanner = new Scanner(System.in);  System.out.println("ATM Menu:"); |
| System.out.println("1. Check Balance"); |
| System.out.println("2. Withdraw"); |
| System  .  out  .  println  (  "3.    Deposit"  )  ; |
| System.out.println("4. Exit"); |
| System.out.print("Enter your choice: "); |
| int choice = scanner.nextInt(); scanner.close(); |
| switch (choice) { |
| case 1: |
| System.out.println("You have chosen to check your |
| balance."); |
| break; |
| case 2: System.out.println("You have chosen to withdraw money."); |
| break; |
| case 3: |
| System.out.println("You have chosen to deposit money."); |

|  |
| --- |
| break; |
| case 4: |
| System.out.println("Exiting..."); |
| break; default: |
| System.out.println("Invalid choice. Please try again."); |
| } |
| } |
| } |



**5.**

**Test**

**Cases**

Present

a

table

of

test

cases

you

used

to

validate

your

program.

Include

a

mix

of

regular,

boundary,

and

edge

cases.

**Test**

**Case**

**No.**

**Input**

**Expected**

**Output**

**Actual**

**Output**

**Status**

**Pass/Fail**

**(**

**)**

**1**

1

You

have

chosen

to

check

your

balance.

You

have

chosen

to

check

your

balance.

pass

**2**

4

exiting

exiting

pass

**3**

6

Invalid

choice.

Please

try

again.

Invalid

choice.

Please

try

again.

pass



**6.**

**Screenshots**

**of**

**Output**

**7.**

**Observation**

**/**

**Reflection**

This

program

demonstrates

a

simple

way

to

use

a

switch

statement

to

handle

menu

selections.

The

switch

statement

makes

the

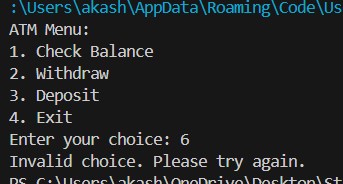
code

more

readable

and

efficient.



**Problem Solving Activity 2.11**

# 1. Program Statement

Grade Remarks using if-else if-else to print and know why switch is not ideal.

# 2. Algorithm

1. Get the score as input.
2. Use if-else if-else to determine the grade remarks based on the score:

- Each condition corresponds to a specific grade range.



**3.**

**Pseudocode**

BEGIN

INPUT

score

IF

score

>=

90

AND

score

<=

100

PRINT

"Excellent"

ELSE

IF

score

>=

80

AND

score

<=

89

PRINT

"Very

Good"

ELSE

IF

score

>=

70

AND

score

<=

79

PRINT

"Good"

ELSE IF score >= 60 AND score <= 69

PRINT "Pass"

ELSE IF score >= 0 AND score < 60

PRINT "Fail"

ELSE

PRINT "Invalid score. Please enter a score between 0 and 100."

END

# 4. Program Code

|  |
| --- |
| import java.util.Scanner; |
| public class D2\_11 { |
|  |
| public static void main(String[] args) { Scanner scanner = new Scanner(System.in); |
| System.out.print("Enter your score (0-100): "); |
| int score = scanner.nextInt(); |
| scanner.close(); |
|  |
| if (score >= 90 && score <= 100) { |
| System.out.println("Excellent");  } else if (score >= 80 && score <= 89) { |
| System.out.println("Very Good"); |
| } else if (score >= 70 && score <= 79) { |
| System.out.println("Good"); |
| } else if (score >= 60 && score <= 69) { |
| System.out.println("Pass"); |
| } else if (score >= 0 && score < 60) {  System.out.println("Fail"); |
| } else { |
| System.out.println("Invalid score. Please enter a score |
| between 0 and 100."); |
| } |
| } |
| } |

# 5. Test Cases

Present a table of test cases you used to validate your program. Include a mix of regular, boundary, and edge cases.



**Test**

**Case**

**No.**

**Input**

**Expected**

**Output**

**Actual**

**Output**

**Status**

**(**

**Pass/Fail**

**)**

**1**

88

Very

Good

Very

Good

pass

**2**

20

Fail

Fail

pass

**3**

104

Invalid

score.

Please

enter

a

score

between

0

and

100

Invalid

score.

Please

enter

a

score

between

0

and

100

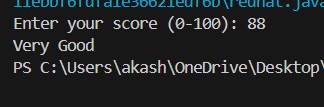
pass

**6.**

**Screenshots**

**of**

**Output**



# 7. Observation / Reflection

A switch statement would not be ideal for this problem because it is designed to handle exact matches, whereas this problem requires handling ranges of values. The if-else if-else structure is more suitable for this problem because it allows us to specify conditions that involve ranges of values.

