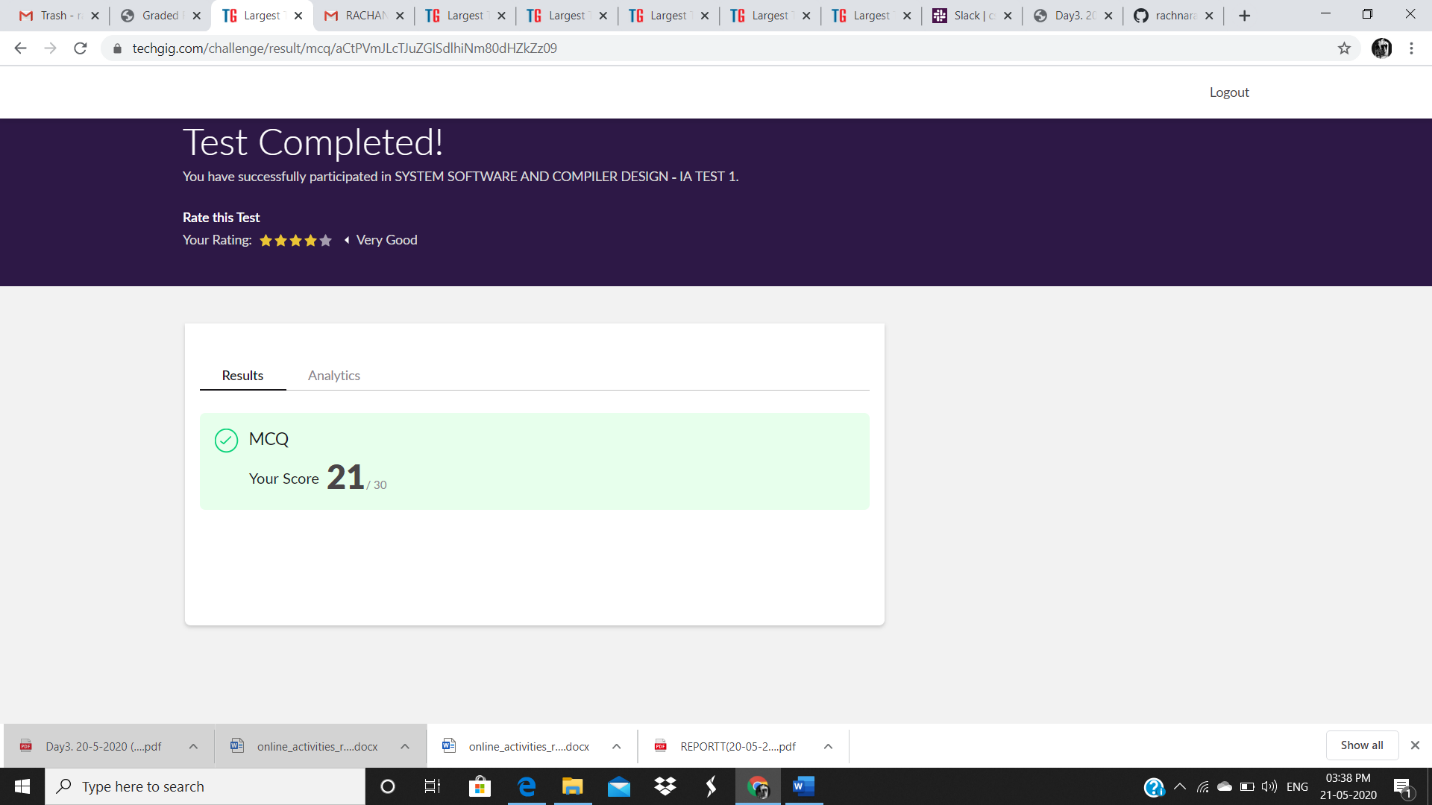
**DAILY ONLINE ACTIVITIES SUMMARY**

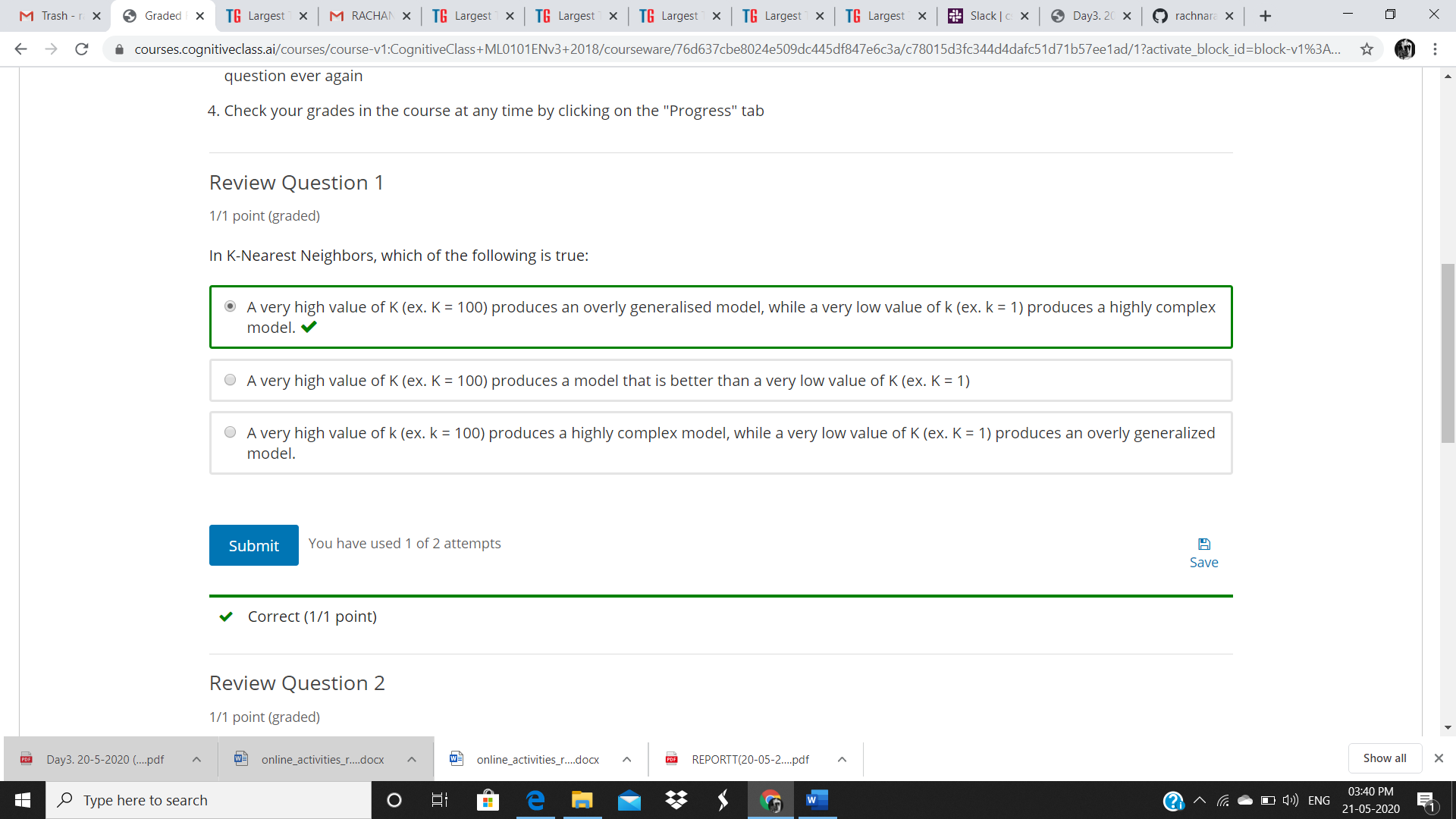
|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Date:** | **20 May 2020** | | | | | **Name:** | **RACHANA K N** | |
| **Sem & Sec** | **6th sem & B sec** | | | | | **USN:** | **4AL17CS070** | |
| **Online Test Summary** | | | | | | | | |
| **Subject** | | **System Software and Compiler Design** | | | | | | |
| **Max. Marks** | | **30** | | **Score** | | | **21** | |
| **Certification Course Summary** | | | | | | | | |
| **Course** | **Machine Learning with python** | | | | | | | |
| **Certificate Provider** | | | **Congnitive Class** | | **Duration** | | | **6 hours** |
| **Coding Challenges** | | | | | | | | |
| **1.Problem Statement:** 1. We have a Letter or a word then we need add some letters to it and need to find out shortest palindrome For example we take "S": S will be the shortest palindrome string. If we take "xyz": zyxyz will be the shortest palindrome string So we need to add some characters to the given string or character and find out what will be the shortest palindrome string by using simple java program  **2.** Write a simple code to identify given linked list is palindrome or not by using stack. First take a Stack. Traverse through each node of the linked list and push each node value to Stack. Once the traversal & copying is done, iterate through linked list from head node again. In each iteration, pop one stack element and compare with node value in respective iteration. It is expected to match stack popped value with node value. In case of all matches, its a palindrome. Any one element mismatch makes it not a palindrome.  **3.** 3.A user will input two strings, and we find if one of the strings is a sub sequence of the other. Program prints “yes” if either the first string is a sub sequence of the second string or the second string is a sub sequence of the first string. Assume that, the length of the first string is smaller than or equal to the length of the second string. | | | | | | | | |
| **Status: DONE** | | | | | | | | |
| **Uploaded the report in Github** | | | | | **YES** | | | |
| **If yes Repository name** | | | | | **Daily Status** | | | |
| **Uploaded the report in slack** | | | | | **YES** | | | |

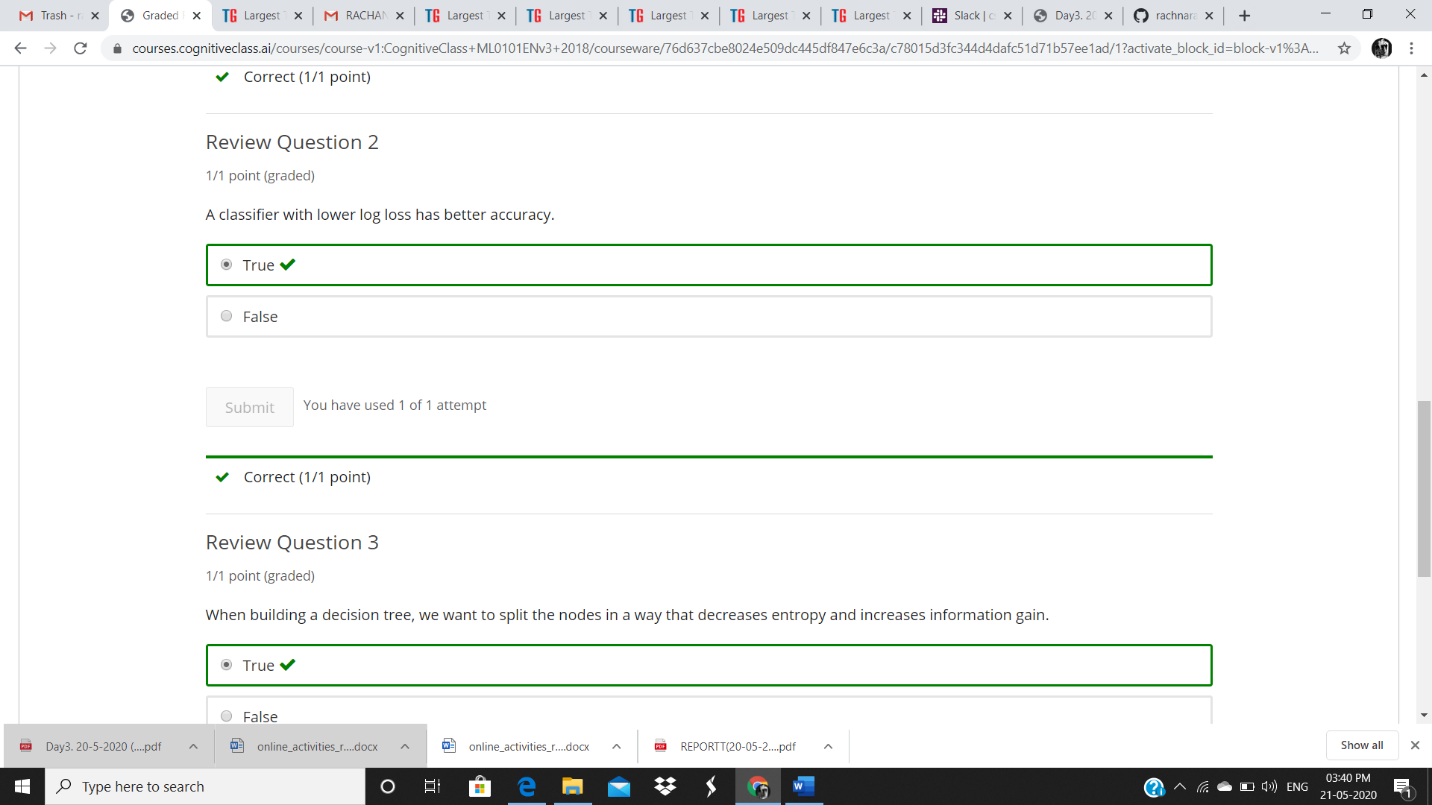
Online Test Details: (Attach the snapshot and briefly write the report for the same)

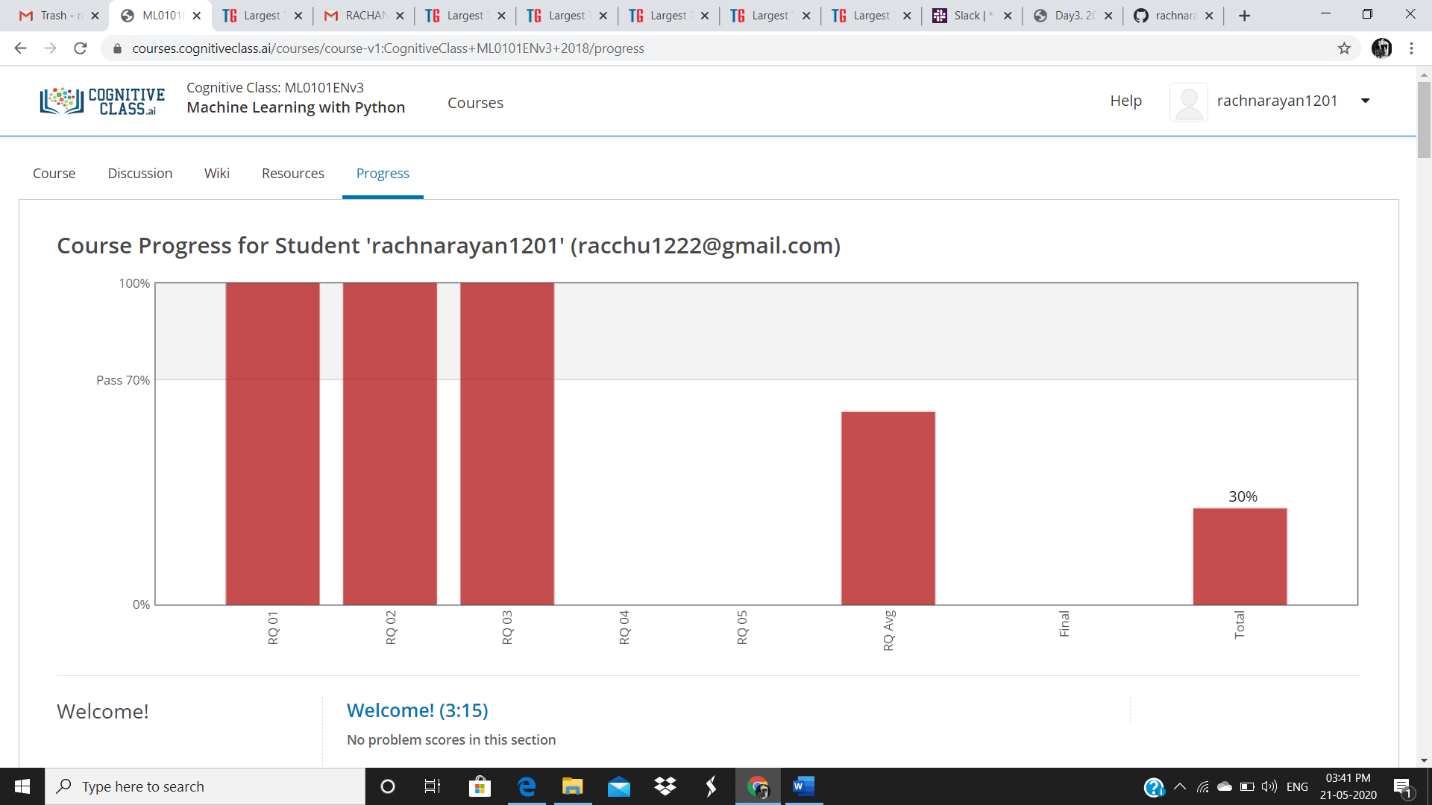
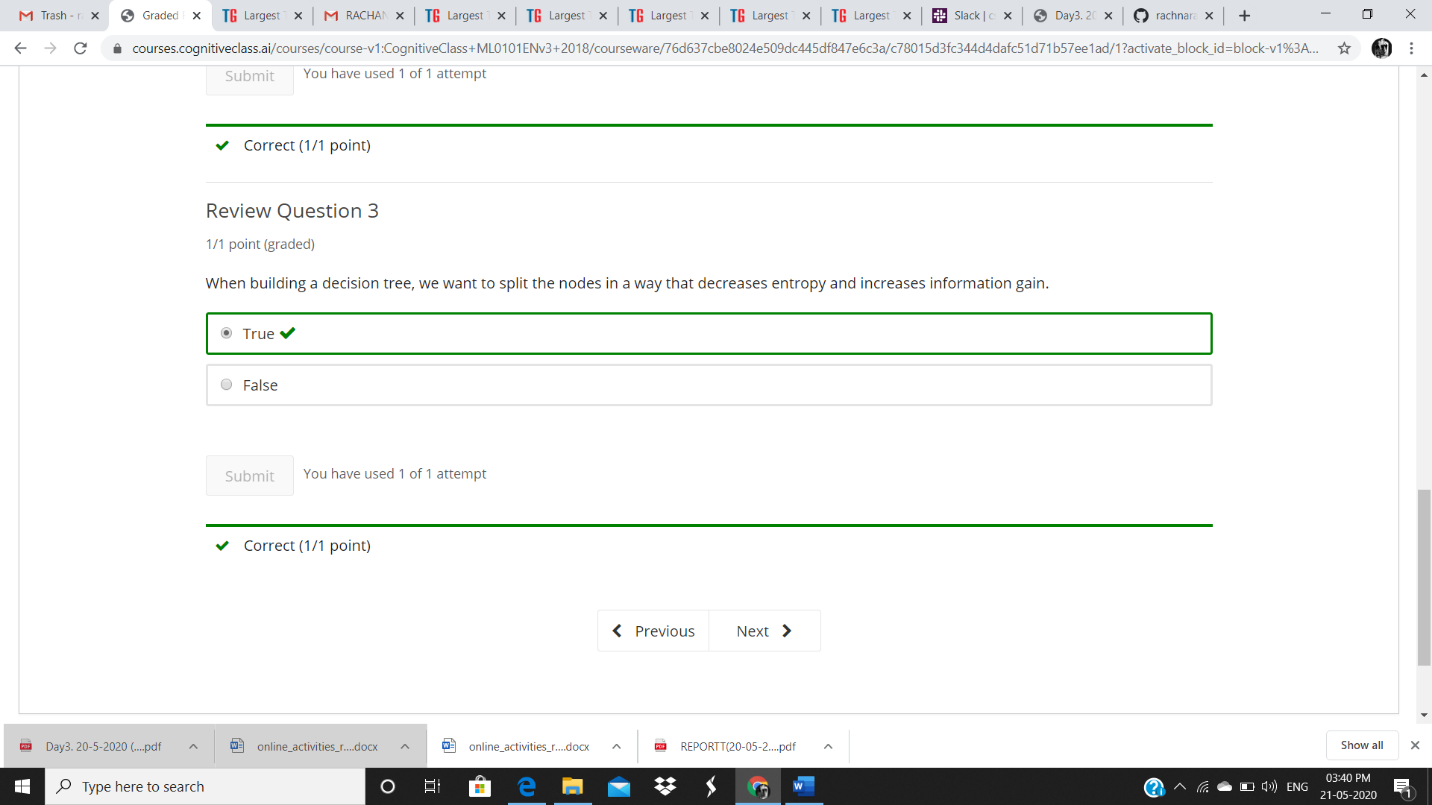


CGV IA test was held today i.e 19 May 2020. There were Three rounds where each round carried marks respectively. Out of 30 marks I scored 21

Certification Course Details: (Attach the snapshot and briefly write the report for the same







DAY 2 (19-05-2020)- Introduction to Regression, MODULE 2 Learning objectives Simple ,Linear ,Non Linear Regression, model Evaluation and Evaluation Metrics AND REVIEW QUESTIONS ARE COMPLETED

Coding Challenges Details: (Attach the snapshot and briefly write the report for the same)

PROGRAM 1:

# Python Program to Reverse a Number

Number = int(input("Please Enter any Number: "))

Reverse = 0

while(Number > 0):

Reminder = Number %10

Reverse = (Reverse \*10) + Reminder

Number = Number //10

print("\nReverse of entered number is = %d" %Reverse)

PROGRAM 2:

Program 2:

#Write a simple Python program to implement Diffie-Hellman Key Exchange Example

# Variables Used

sharedPrime = int(input("\nEnter the value of p(shared prime)")) #23

sharedBase = int(input("\nEnter the value of g(shared base)")) #5

aliceSecret = int(input("\nEnter the value of a(alice secret)")) #6

bobSecret = int(input("\nEnter the value of a(bob secret)")) #15

# Begin

print( "\nPublicly Shared Variables:")

print( " Publicly Shared Prime: " , sharedPrime )

print( " Publicly Shared Base: " , sharedBase )

# Alice Sends Bob A = g^a mod p

A = (sharedBase\*\*aliceSecret) % sharedPrime

print( "\n Alice Sends Over Public Chanel: " , A )

# Bob Sends Alice B = g^b mod p

B = (sharedBase \*\* bobSecret) % sharedPrime

print( " Bob Sends Over Public Chanel: ", B )

print( "\n------------\n" )

print( "Privately Calculated Shared Secret:" )

# Alice Computes Shared Secret: s = B^a mod p

aliceSharedSecret = (B \*\* aliceSecret) % sharedPrime

print( " Alice Shared Secret: ", aliceSharedSecret )

# Bob Computes Shared Secret: s = A^b mod p

bobSharedSecret = (A\*\*bobSecret) % sharedPrime

print( " Bob Shared Secret: ", bobSharedSecret )