

**Programming I**

Year 1 (2021/22), Semester 1

## SCHOOL OF INFOCOMM TECHNOLOGY

Diploma in Cybersecurity & Digital Forensics

Diploma in Data Science

Diploma in Information Technology

Diploma in Immersive Media

Common ICT Programme

**Revision Paper**

Date:

Time:

INSTRUCTIONS TO CANDIDATES:

1. Write your Student Number, Name, Module Group and Seat Number CLEARLY in the boxes provided above.
2. This paper consists of \_\_ pages including this cover page. Check carefully to make sure your set is complete
3. There are 4 questions. Answer ALL questions.
4. You are allowed to refer to the hardcopy or softcopy of PRG1 teaching materials and programs
5. No electronic devices except calculator are permitted.

**QUESTION 1** (20 marks)

Tom is training for a swimming competition and records the time taken for each of 2 laps.

Tom has written a Python program to help him find his best and average times. He has made a couple of mistakes in his program. There may be more than one mistake per line. Help Tom debug and resolve the errors.

The Python program that Tom has created is shown below.

|  |
| --- |
| times = input('Enter time taken of 2 laps separated by semi-colon (seconds):'  times\_list = times.split(';')  firstlap\_time = times\_list[0]  secondlap\_time = times\_list[1]  if firstlap\_time > secondlap\_time:  best = firstlap\_time  else:  best = secondlap\_time  total = firstlap\_time + second\_time  print('Tom's best time is {:.1f} s and average time is {total} s'.format(best)) |

Create a new .py file with IDLE, and type the codes into the .py file. Correct the program, then copy and paste the corrected code into the answer space below.

The following is a sample output of the corrected program (values underlined are the user’s input):

Enter time taken of 2 laps separated by semi-colon(seconds): 2;3

Tom's best time is 2.0 s and average time is 2.5 s

(20 marks)

|  |
| --- |
| times = input('Enter time taken of 2 laps separated by semi-colon (seconds):')  times\_list = times.split(';')  times\_list = [float(i) for i in times\_list]  firstlap\_time = times\_list[0]  secondlap\_time = times\_list[1]  if firstlap\_time < secondlap\_time:  best = firstlap\_time  else:  best = secondlap\_time  total = (firstlap\_time + secondlap\_time) / 2  print("Tom's best time is {:.1f} s and average time is {:.1f} s".format(best,total)) |

**QUESTION 2** (30 marks)

In a card game, the cards 2 to 10 are counted at their face values, regardless of suit (clubs, diamonds, hearts or spades). All face cards (jack, queen and king) are counted as 10. An ace is counted as a 1 or an 11, depending on the total count of all cards in a player’s hand. The ace is counted as 11 only if the resulting total value of all cards does not exceed 21. Otherwise, it is counted as 1.

A program will be used to calculate the total count in a player’s hand. The program will prompt the user to enter three card values, calculate the total value of the hand and display the value of the three cards. An input of 1 corresponds to an ace, an input of 2 corresponds to a two, …, an input of 11 corresponds to a jack, an input of 12 corresponds to a queen, and so on.

Two sample runs of the program are as follows:

Enter card 1: 5

Enter card 2: 6

Enter card 3: 1

Total value is 12

Enter card 1: 12

Enter card 2: 1

Enter card 3: 1

Total value is 12

Lists and other data structures are not allowed for this program.

1. Write the pseudocode for the program in the space provided below.

(15 marks)

|  |
| --- |
| 1. Input card 1 2. Input card 2 3. Input card 3 4. Check if card 1 is above 10, change to 10 if it is 5. Check if card 2 is above 10, change to 10 if it is 6. Check if card 3 is above 10, change to 10 if it is 7. Check if card 1 is 1, change to 11 if the total of 3 cards is below 21 8. Check if card 2 is 1, change to 11 if the total of 3 cards is below 21 9. Check if card 3 is 1, change to 11 if the total of 3 cards is below 21 10. Find the total sum of the 3 cards after checking 11. Display the total value of the cards |

**QUESTION 2** (cont.)

1. Write the program in Python.

(15 marks)

|  |
| --- |
| a = int(input('Enter card 1:'))  b = int(input('Enter card 2:'))  c = int(input('Enter card 3:'))  if a > 10:  a = 10  elif b > 10:  b = 10  elif c > 10:  c = 10  d=a+b+c  if a == 1 or b==1 or c==1:  d += 10  if d>21:  d-=10  print('Total value is:', d)  or  a = int(input('Enter card 1:'))  b = int(input('Enter card 2:'))  c = int(input('Enter card 3:'))  if a > 10:  a = 10  elif b > 10:  b = 10  elif c > 10:  c = 10  elif a == 1:  a = 11  d = a + b + c  if d <= 21:  a = 11  else:  a = 1  elif b == 1:  b = 11  d = a + b + c  if d <= 21:  b = 11  else:  b = 1  elif c == 1:  c = 11  d = a + b + c  if d <= 21:  c = 11  else:  c = 1  d = a + b + c  print('Total value is:', d) |

**QUESTION 3** (30 marks)

In an experiment to find the relationship between a student’s BMI and his t-shirt size, the following data were collected:

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Height(cm) | Weight(kg) | Size |
| Sharon | 172 | 59.5 | M |
| Mic | 166 | 65.6 | L |
| Josh | 187 | 49.8 | S |
| Hannah | 200 | 64.2 | M |
| Hanzel | 166 | 47.5 | S |
|  |  |  |  |

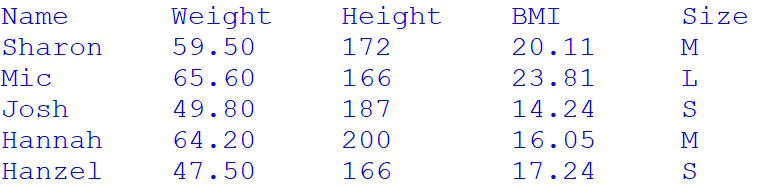
(Assume that only 3 sizes: S, M and L are being considered)

1. Using Python, create 4 lists (name\_list, height\_list, weight\_list, size\_list) to store the data given in the table, one list for each column.

Create another new list (bmi\_list), to calculate and store the student’s corresponding BMI.

(The formula for calculating BMI is weight/(height/100)2)

Print a table using the 5 lists, to present the data as follows:



(8 marks)

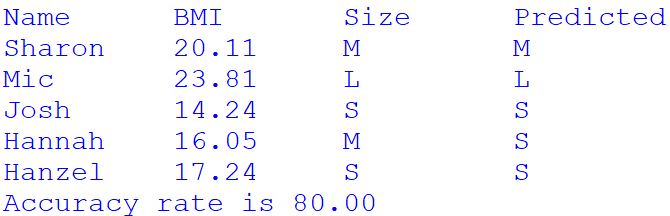
1. It is proposed that the following guideline be used to predict a student’s t-shirt size based on his BMI:

|  |  |
| --- | --- |
| bmi | size |
| Less than or equal to 18 | S |
| Between 18 and 21 inclusive | M |
| More than 21 | L |

Create a list (predicted\_list), to predict based on the above guideline and store the student’s t-shirt size.

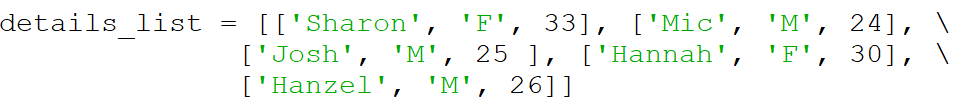
Print a table using the lists created in part 3(a) and part 3(b) , followed by the accuracy rate of the prediction, which is the percentage of correct predictions, in 2 decimal places.

You do not need to include the code for creation of lists in part 3(a). BMI should not be re-calculated again.



(12 marks)

1. Assuming that a nested list, containing the following information (name, gender, age) was given.



Given weight, height, gender and age, the BMR (Basal Metabolic Rate) of a student can be calculated:

BMR for Men = 66.47 + (13.7 \* weight) + (5 \* height) − (6.8 \* age)

BMR for Women = 655.1 + (9.6 \* weight) + (1.8 \* height) − (4.7 \* age)

Print a table using the lists created in QUESTION part 3(a) and the details\_list to present the following information. You do not need to include the code for creation of lists in part 3(a).



(10 marks)

|  |
| --- |
| # 3a  name\_list = ['Sharon', 'Mic', 'Josh', 'Hannah', 'Hanzel']  height\_list = [172, 166, 187, 200, 166]  weight\_list = [59.5, 65.6, 49.8, 64.2, 47.5]  size\_list = ['M', 'L', 'S', 'M', 'S']  a = 0  BMI\_list=[]  print('{:<10}{:<10}{:<10}{:<10}{:<10}'.format('Name', 'Weight', 'Height', 'BMI', 'Size'))  while a < 5:  BMI = (weight\_list[a] / ((height\_list[a] / 100) \*\* 2))  BMI\_list.append(BMI)  print('{:<10}{:<10.2f}{:<10}{:<10.2f}{:<10}'.format(name\_list[a], weight\_list[a], height\_list[a], BMI\_list[a], size\_list[a]))  a = a + 1  # 3b  predicted\_list = ['M', 'L', 'S', 'S', 'S']  b = 0  acc = 0  print('\n{:<10}{:<10}{:<10}{:<10}'.format('Name', 'BMI', 'Size', 'Predicted'))  while b < 5:  BMI = (weight\_list[b] / ((height\_list[b] / 100) \*\* 2))  print('{:<10}{:<10.2f}{:<10}{:<10}'.format(name\_list[b], BMI, size\_list[b], predicted\_list[b]))  if size\_list[b] == predicted\_list[b]:  acc = acc + 1  b = b + 1  acc = (acc / len(predicted\_list)) \* 100  print('Accuracy rate is {:0.2f}'.format(acc))  # 3c  c = 0  Gender = ['F', 'M', 'M', 'F', 'M']  AGE = [33, 24, 25, 30, 26]  print('{:<15}{:<15}{:<15}{:<15}{:<15}{:<15}{:<15}'.format('Name', 'Weight', 'Height', 'BMI', 'Gender', 'Age', 'BMR'))  while c < 5:  BMI = (weight\_list[c] / ((height\_list[c] / 100) \*\* 2))  if Gender[c] == 'M':  BMR = 66.47 + (13.7 \* weight\_list[c]) + (5 \* height\_list[c]) - (6.8 \* AGE[c])  else:  BMR = 655.1 + (9.6 \* weight\_list[c]) + (1.8 \* height\_list[c]) - (4.7 \* AGE[c])  print('{:<15}{:<15.2f}{:<15}{:<15.2f}{:<15}{:<15}{:<15.2f}'.format(name\_list[c], weight\_list[c], height\_list[c], BMI,Gender[c], AGE[c], BMR))  c = c + 1 |

**QUESTION 4** (20 marks)

Triangular Numbers are a series of numbers that can be represented in the form of a triangular grid of points where the first row contains a single element and each subsequent row contains one more element than the previous one as shown in the figure below:

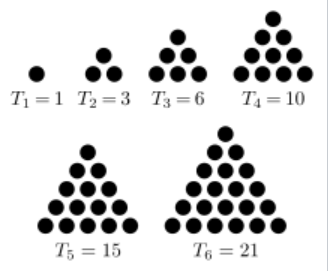


Figure 1: the first 6 triangular numbers (1, 3, 6, 10, 15, 21)

The *n*th triangular number is the number of dots in the triangular arrangement where *n* is the number of dots on the base of the triangle. The formula to calculate the nth triangular number, Tn is:

Tn = ((n\*\*2) + n)/2

Your task is to write a Python program that:

1. Ask the user to enter a number between 0 and 5000.
2. If the number entered is a triangular number, display the value of n
3. Otherwise, display the message that the number entered is not a triangular number

You are required to use loop(s) in your program.

Sample outputs:





|  |
| --- |
| count=0  import math  while count<3:  a = int(input('Enter a number between 0 and 5000:'))  b = a \* -2  c = ((1 + math.sqrt(1 - 4 \* b)) / 2) - 1  if c == int(c):  print('{:.0f} is a triangular number and it is the {:.0f}th number.'.format(a,c))  else:  print(a, 'is NOT a triangular number.')  count=count+1 |

\*\* END OF PAPER \*\*\*