|  |  |
| --- | --- |
| **Programming 1 (PRG1)**  Diploma in IT / DS / CSF / IM / CICTP  Year 1 (2023/24) Semester 1 | Week **4** |
|  |
| **Exercise 4: Selection Structure** | |

**OBJECTIVES**

At the end of this exercise, students should be able to develop Python programs involve:

* if statement
* if … else statement
* if … elif … else statement

**IMPORTANT**

* Create a folder, **Week04**, in your hard disk.
* For programming questions, create Python programs with the given file names in the **Week04** folder created above. Do add the description, your name and student ID as comments at the beginning of each program.
* For non-programming questions, type your answers in the boxes provided below the questions.
* At the end of the session, compress all the files in your **Week04** folder (i.e. the Python program files and this word document) and submit the zip file in POLITEMall.

**Part 1**

Activity 1

Calculate Commission ( file name: CalComm.py )

* ABC Company pays its sales agents on a commission basis. Each agent is paid 10% commission for monthly sales above or equal to $10,000 and 5% commission for monthly sales below $10,000.
* Write a program to accept the monthly sales of a particular agent and based on the amount, determine and print out the commission earned.
* Sample output (values underlined are the user input):

|  |
| --- |
| Enter monthly sales of sales agent: 12000  The commission rate is : 10%  The commission paid is : $1200.00 |

|  |
| --- |
| Enter monthly sales of sales agent: 8000  The commission rate is : 5%  The commission paid is : $400.00 |

Activity 2

Sum Two Random Numbers ( file name: SumRandomNumbers.py )

* Write a program that generates two integers between 0 and 100 inclusive and prompts the user to enter the sum of these 2 integers. The program reports if the answer is correct or wrong (program will also print the correct answer if the user answer is wrong).
* Hint: you need to use the random module

i.e. import random

num1 = random.randint(0,100)

* Sample output(values underlined are the user input):

|  |
| --- |
| Enter the sum of 45 and 35: 51  Your answer is wrong.  The correct answer is 80. |

|  |
| --- |
| Enter the sum of 93 and 6: 99  Your answer is correct! |

Activity 3

Determine Grade – ( file name: Grades.py )

* The grade that you get for a module depends on your marks. Table below shows the grade for each range of marks.

|  |  |  |
| --- | --- | --- |
| Marks | Grade | Comment |
| >= 85 | A+ | Excellent! |
| >=80 and < 85 | A | Well done. |
| >=75 and < 80 | B+ | - |
| >=70 and < 75 | B | - |
| >=65 and < 70 | C+ | - |
| >=60 and < 65 | C | - |
| >=55 and < 60 | D+ | - |
| >=50 and < 55 | D | - |
| < 50 | F | See me. |

* Write a program that asks for a student’s marks, then displays the correct grade and the corresponding comment (if any).
* Sample output (values underlined are the user input):

|  |
| --- |
| Please enter your marks: 83  Grade: A  Well done. |

|  |
| --- |
| Please enter your marks: 54.3  Grade: D |

|  |
| --- |
| Please enter your marks: 33  Grade: F  See me. |

Activity 4

Students are asked to write a code snippet to print the health risk depending on the value in bmi. John has written the code shown in figure 1 and Tom has written the code shown in figure 2. However, the teacher said that both the codes are inefficient. Could you explain to John and Tom why their codes are inefficient.

|  |
| --- |
| if bmi < 18.5:  print("Risk of developing osteoporosis")  elif 18.5 <= bmi < 23:  print("Healthy")  elif 23 <= bmi < 27.5:  print ("Low risk of developing heart disease, stroke, etc.")  elif bmi >= 27.5:  print("High risk of developing heart disease, stroke, etc.") |
| Figure 1 |
| if bmi < 18.5:  print("Risk of developing osteoporosis")  if 18.5 <= bmi < 23:  print("Healthy")  if 23 <= bmi < 27.5:  print ("Low risk of developing heart disease, stroke, etc.")  if bmi >= 27.5:  print("High risk of developing heart disease, stroke, etc.") |
| Figure 2 |

Answer:

|  |
| --- |
| **For the last line, he can use else instead of elif/if.** |

Activity 5

Guard in Computer Game – ( file name: GuardAI.py )

* A guard in a computer game acts as follows:
  + If sees\_player == False, he will wait
  + If sees\_player == True
    - If dist\_from\_player <= 1, he will attack
    - If 2 <= dist\_from\_player <= 4, he will advance
    - If dist\_from\_player >= 5, he will wait
* Write a program that shows how the guard will act.
* Sample output(values underlined are the user input):

|  |
| --- |
| Does the guard see the player (y/n)? n  The guard waits. |

|  |
| --- |
| Does the guard see the player (y/n)? y  How far away is the player? 1  The guard attacks! |

|  |
| --- |
| Does the guard see the player (y/n)? y  How far away is the player? 7  The guard waits. |

|  |
| --- |
| Does the guard see the player (y/n)? y  How far away is the player? 3  The guard advances. |

**Part 2**

Activity 1

Write compound conditions for the following with suitable variable names

|  |  |
| --- | --- |
| Temperature is less than or equal to 25, or humidity is less than 70% | Temperature <= 25 or Humidity <= 0.7 |
| Income between $4000 and $6000 inclusive | Income >=4000 and <=6000 |
| Either a Mage, or a Sorcerer who is at least level 10 | (Prof == “Mage” or prof == “Sorcerer”) and level >= 10 |
| First year students who are either scholarship holders or has GPA of 4 | Year == 1 and (scholarship == True or GPA == 4) |
| Person is older than 18 or female, but not both | (age >= 18 or gender != ‘F’) |

Activity 2

Determine Leap Year – ( file name: DetLeapYear.py )

* A year is said to be a leap year if it is divisible by 4 but not divisible by 100, except those that is divisible by 400.
* Write a program to determine if an input year is a leap year or not.
* Sample output(values underlined are the user input):

|  |
| --- |
| Please enter the year: 2000  2000 is a leap year. |

|  |
| --- |
| Please enter the year: 2016  2016 is a leap year. |

|  |
| --- |
| Please enter the year: 2018  2018 is not a leap year. |

|  |
| --- |
| Please enter the year: 1800  1800 is not a leap year. |

Activity 3

Validate password – ( file name: ValidatePassword.py )

* A valid password must be at least 8 characters long and contains at least one uppercase letter, one lowercase and one digit.
* Write a program to determine if an input password is a valid and print the result from the validation.
* Sample output (values underlined are the user input):

|  |
| --- |
| Enter password: abcde  Password must be at least 8 characters long. |

|  |
| --- |
| Enter password: Abcdefgh  Password must contain at least one digit. |

|  |
| --- |
| Enter password: abcd1234  Password must contain at least one uppercase letter. |

|  |
| --- |
| Enter password: Abcd1234  Password is valid. |

* Hint: you may like to use the search() method in the Python “re” module.

**OPTIONAL**

Create a data file contain your weight in kilogram, your height in meter, your gender and your age in one line, separate with commas. Save the file as “data.txt” in the same folder as your Python program.

Example of data.txt:

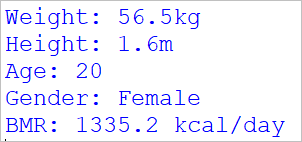
|  |
| --- |
| 56.5,1.65,Female,20 |

Write a Python program (save as BmrCalculator.py) to read the data from the data file, calculate and print the Basal Metabolic Rate (BMR).

Note:

Formula to calculate BMR for men: **10 \* weight in kg + 6.25 \* height in cm – 5 \* age + 5**

Formula to calculate BMR for woman: **10 \* weight in kg + 6.25 \* height in cm – 5 \* age -161**

Sample output: