

Programming in Python

by Philip Richmond

Submission date: 18-Jan-2023 07:05PM (UTC+0000)

Submission ID: 195512099

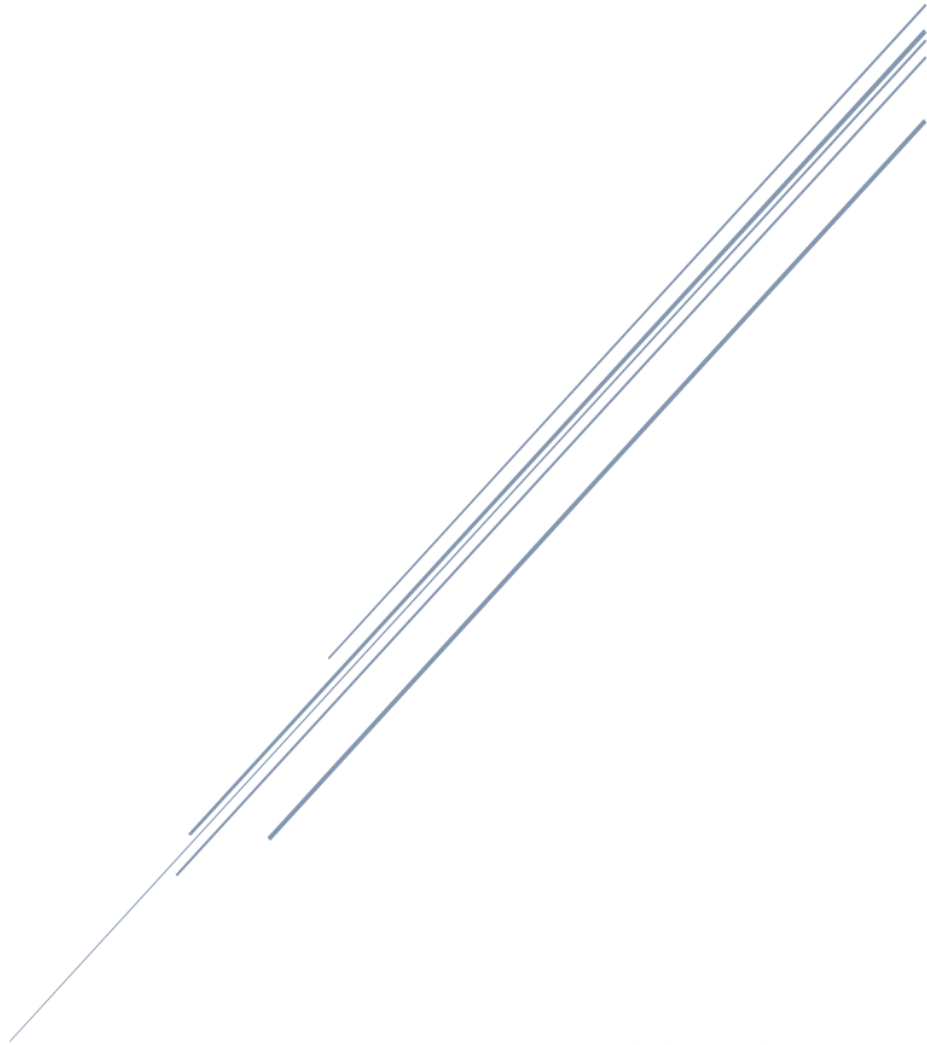
File name: 41405_Philip_Richmond_Programming_in_Python_466019_1667178333.docx (168.75K)

Word count: 2119

Character count: 10692

PROGRAMMING IN PYTHON

ACCA4020



By Philip Richmond
2212266

Table of content

PART A (UNDERSTANDING)

A1	Page 2
A2	Page 2
A3	Page 2 / 3

PART B (IMPLEMENTATION)

Exercise 1:	Page 3 / 4
Exercise 2:	Page 5 / 6
Exercise 3:	Page 7 / 8
Exercise 4:	Page 9 / 10
Conclusion	Page 11
References	Page 12

Part A:

A1: A variable is a named location in python, s memory where I can store, retrieve and change data. Variables are important in writing programs as the programmer can store these values and call them back at any time through the program and makes it easier for the programmer to modify and maintain the program.

The different types of variables that we have used are.

- **“int / integer”** which represents whole numbers (e.g., 1, 2, 3, 4,)
- **“float / floating”** which represents a decimal number (e.g., 1.0, 2.5, 3.22, 4.05,)
- **“str / string”** which represents a sequence of characters (e.g., “Hello I would like a cup of tea please”)

A2: “if/else” statements are a way the programmer can allow a program to execute different code paths to his desired outcome. Show below is an example I made.

```
print("How old are you")
userAge = int(input())

if userAge >= 18:
    print("You are a adult")
else:
    print("You are a adolescent")
print("End of program")
```

In this code “if” checks the user input against the value to determine whether the inputted age is greater than 18 and if it is it will print the statement “You are a adult”. Any other age below 18 will trigger a statement from “else: printing “You are a adolescent”.

A3: Loops are a fundamental part of programming that allows the programmer to repeat a set of instructions until a certain condition has been met. This is handy for a variety of purposes like making lists of items or repeating a part of code in till the input meets the desired outcome.

An example of a program that uses lists is below

```
import random

numberToGuess = random.randint( 1 , 10 )

while True:
    userGuess = int(input("Guess a number between 1 and 10: "))

    if userGuess == numberToGuess:
        print("Congratulations! You guessed the correct number.")
        break
```

```
elif userGuess < numberToGuess:
    print("The number you guessed is too low. Try again.")
else:
    print("The number you guessed is too high. Try again.")
```

Breaking out of loops

The Author found out that [The Python break keyword can be used to prematurely terminate a loop when a specified condition is met. The break statement is situated inside the loop statement block and is preceded by a test expression. When the test returns True, the loop ends immediately, and the program proceeds on to the next task. For example, in a nested inner loop it proceeds to the next iteration of the outer loop] [1]

Part B:

Exercise 1:

The author's code below is a simple script that prompts the user for various pieces of information: their name, national insurance number, age and monthly take-home pay. It then checks the age to ensure it is greater than 15, and the monthly take-home pay to ensure it is less than 5000. This is to show that if the input is too high that it will ask again to make sure it's correct. If the age is less than 15, the user is prompted to try again showing that the age is too young for the program. Once all the input is valid, the program ends and thanks the user.

```
print("Please enter your name:")
userName = str(input())

print("Please enter your NI number:")
userNi = str(input())

print("Please enter your age:")
userAge = int(input())
while userAge <= 15:
    print("That's too young:")
    userAge = int(input("Try again:"))

print("Please enter your monthly take home pay:")
userPay = int(input())
if userPay >= 5000:
    print("Wow is that right")
    userPay = int(input("Try Again:"))

print("Thank you end of program")
```

Key point to program.

- 1: The program uses the built-in '**input()**' function to prompt the user for input and store the input in variables ('userName', 'UserNi' , 'userAge' , 'userPay')
- 2: The program uses the '**str()**' and '**int()**' function to convert the input from '**input()**' into the appropriate data types (string and integer)
- 3: The program uses a '**while**' loop to repeatedly prompt the users for thire age until a valid age (greater than 15) is entered.
- 4: The program uses and '**if**' statement to check if the users monthly take-home pay is greater than or equal to 5000 and, if so, prompts the user to try again.
- 5: The program uses the '**print()**' function to display messages to the user, such as prompts for input and thank you message at the end of the program.

[2]

Variable Types:-		
Name	Type	Description
Integers	int	Integers ,like 2,-3.
Float	float	Decimal number, like 3.2,5.2.
String	str	Ordered sequence of characters, like "Intellify".
List	list	Ordered sequence of various data types, like [1,'U',3,5.2].
Dictionary	dict	Unordered sequence of Key-Value pair, lke {"a":1, "b".2}.
Tuples	tup	Unmutable sequence of data types, like (1,3.2,'a').
Set	set	Collection of unique elements, like ('a','b').
Boolean	bool	Logical values indicating True or False only.

www.intellify.in RITIK, UG JCBUST

Exercise 2:

The author code below shows a program with a function called `subtractSquareFromCircle()` that prompts the user for radius of a circle and side of a square. It uses input to calculate the area of the circle and the square. The program then enters a while loop where it checks if the area of the square is greater than the area of the circle, if yes it prompts the user to enter a smaller square size number. Once the square is smaller than the circle area, the program calculates the remaining area of the circle after the square is subtracted and prints it out.

```
def subtractSquareFromCircle():
    circleRadius = float(input("Enter the radius of the circle in cm: "))
    # the circle size from user
    circleDiameter = 2 * circleRadius
    circleArea = 3.14 * (circleRadius ** 2)

    squareSide = float(input("Enter the side length of the square in cm: "))
    # The square size from user
    squareArea = squareSide ** 2

    while squareArea > circleArea: # does the square fits in the circle
        print("The square is to big. Please enter a smaller square.")
        squareSide = float(input("Enter the side length of the square: "))
        squareArea = squareSide ** 2

    remainingCircleArea = circleArea - squareArea # Calculate remaining
    area of circle after square is subtracted
    print("The remaining area of the circle is:", remainingCircleArea)

subtractSquareFromCircle()
```

Validation of program

Enter the radius of the circle in cm: 10
Enter the side length of the square in cm: 700
The square is to big. Please enter a smaller square.
Enter the side length of the square: 7
The remaining area of the circle is: 265.0

Process finished with exit code 0

Key point to program.

- 1: The program uses the `input()` function to get the input from user.

- 2: The program uses the `float()` function to convert the input to float data type.
- 3: The program uses mathematical operations (multiplication and exponentiation) to calculate the area of the circle and the square.
- 4: The program uses a **while** loop to check if the square fits in the circle and prompts the user to enter a smaller square if it does not fit.
- 5: The program uses the print statement to output the remaining area of the circle with the square subtracted.

Exercise 3:

The program below is a shopping list application where the user can input items and their costs. The items and costs are stored in a list and the total cost is calculated as the items are added. The program also includes a bonus discount feature where a random discount between 10% and 40% is applied to the total cost, and the final cost is calculated and displayed.

```
import random

shoppingList = [] # make an empty list

totalCost = 0 # create a variable to store total cost

def add_item(item, cost): # function add items and cost to the list
    shoppingList.append({"item": item, "cost": cost})
    global totalCost
    totalCost += cost

while True: # loop to keep adding items to the list
    item = input("Enter the name of the item you want to add(or 'done' if you finished): ")
    if item == "done":
        break
    cost = float(input("Enter the price of the item in £: "))
    add_item(item, cost)

print("\nHere's your shopping list:") # show the list and the total cost
for item in shoppingList:
    print(item["item"] + " £" + "{:.2f}".format(item["cost"]))
print("The total cost is £" + "{:.2f}".format(totalCost))

bonusDiscount = random.randint(10,40) # get the bonus discount

print("Congratulations! You have won a bonus discount of " +
      str(bonusDiscount) + "%!") # make the bonus discount

finalCost = totalCost - (totalCost * (bonusDiscount/100)) # calculate
final cost after the bonus discount

print("The final cost after the bonus discount is £" +
      "{:.2f}".format(finalCost)) # print the final cost
```

Key point to program.

- 1: The program creates a shopping list application where the user can input and their corresponding costs.

- 2: The items and costs are stored in a list and the total cost is calculated as the items are added
- 3: The program includes a bonus discount feature where a random discount between 10% and 40% is applied to the total cost.
- 4: The final cost after the bonus discount is calculated and displayed
- 5: The program uses a **while** loop to repeatedly prompt the user to input items and costs and a function to add the item and costs to the list and update the total cost.
- 6: The program uses the format function to display the cost in a specific format.
- 7: The mathematical calculations in the program are addition and subtraction to calculate total cost and final cost after discount.
- 8: The program uses the **randint** function from random library to generate a random number between 10 and 40 as a bonus discount.

Validation of program

Enter the name of the item you want to add(or 'done' if you finished): beef
Enter the price of the item in £: 10
Enter the name of the item you want to add(or 'done' if you finished): gravy
Enter the price of the item in £: 2
Enter the name of the item you want to add(or 'done' if you finished): sprouts
Enter the price of the item in £: 3.22
Enter the name of the item you want to add(or 'done' if you finished): potatoes
Enter the price of the item in £: 3
Enter the name of the item you want to add(or 'done' if you finished): done

Here's your shopping list:

beef £10.00
gravy £2.00
sprouts £3.22
potatoes £3.00

The total cost is £18.22

Congratulations! You have won a bonus discount of 14%!

The final cost after the bonus discount is £15.67

Process finished with exit code 0

Exercise 4:

The program below uses the turtle library in python to draw the word “Zoo” on the screen by controlling a turtle pointer. The turtle pointer, called `myZoo` is created at the start of the program and is then used to move around the screen and draw the letters.

```
import turtle

myZoo = turtle.Turtle()

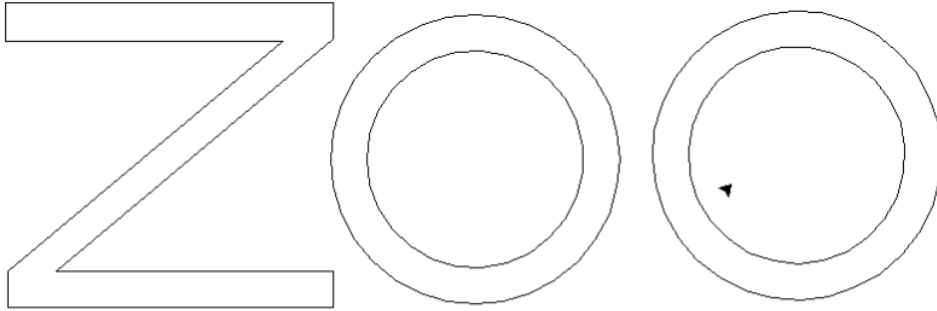
myZoo.penup()
myZoo.backward(+300)
myZoo.pendown()
myZoo.forward(200)
myZoo.right(90) # first corner 140
myZoo.forward(30)
myZoo.left(90)
myZoo.right(140)
myZoo.forward(300)
myZoo.left(140)
myZoo.forward(230)
myZoo.right(90)
myZoo.forward(30)
myZoo.right(90)
myZoo.forward(270)
myZoo.right(90)
myZoo.forward(30)
myZoo.left(-50)
myZoo.forward(297)
myZoo.left(140)
myZoo.forward(230) #
myZoo.right(90)
myZoo.forward(32) #
myZoo.right(90)
myZoo.forward(90)

for secondCircle in range (0, 2):

    myZoo.penup() # move from z
    myZoo.forward(180)
    myZoo.right(90)
    myZoo.forward(130) #
    myZoo.pendown() # start of circles
    myZoo.circle(120)
    myZoo.left(90)
    myZoo.penup()
    myZoo.forward(30)
    myZoo.right(90)
    myZoo.pendown()
    myZoo.circle(90)
    myZoo.left(90)
    myZoo.penup()
    myZoo.forward(30)
    myZoo.right(-24)
    myZoo.pendown()

turtle.done()
```

Validation of program in Turtle



Conclusion

In conclusion the above code of different programs demonstrates a variety of task using different Python constructs such as functions, loops, and conditional statements.

Exercise 1 prompts the user for several pieces of information: name, NI number, age and monthly take-home pay. It stores the users input in variables and make sure that the age is greater than 15 and pay is less than 5000, it will not allow the user to proceed until the correct input is provided. The user also considers make a variable that will show when entering the NI number that a sequence of too letters, five number and finally one letter will have to be correctly entered.

Exercise 2 defines a function called "subtractSquareFromCircle()" which was a bit challenging and frustrating due to the math involved, it accepts the radius of the circle and the side length of a square from the user, checks if the square is fitting in the circle and finds the remaining area of the circle after subtracting the square from it.

Exercise 3 creates a list application here the use can add items to a Christmas shopping list along with their cost and calculates the final coast after a random bonus discount.

The 4 exercise is using the turtle library to create a drawing on the screen of the word Zoo also the program makes use of a for loop which is used to draw two circles. The users found that exercise 4 was the most enjoyable exercise to do.

References

[1] M. McGrath, "making statements," in *Python in easy steps*, 2nd Edition ed., p. 58.

[2] U. J. RITIK, 20 december 2022. [Online]. Available:
https://www.youtube.com/watch?v=r3Q9C_THLRQ.

Programming in Python

GRADEMARK REPORT

FINAL GRADE

62/100

GENERAL COMMENTS

Instructor

PART A

A1 Variables - You have talked about variables and types, but not about how types are inferred in Python. You could also have mentioned that in Python, types can change within the lifetime of the program, for example, when dividing two integers, the resulting type is changed to float)

2.5 /5

A2 If/Else - Explanation is OK but a bit brief. You could also have mentioned multiple conditions and if/elif/else. Example is good.

3.5/5

A3 Loops - You've said what a loop is, but you need to state the different types (for and while) and that for loops are counted, so the number of iterations is known before the loop starts (as opposed to a while loop where it is not known.)

It's not really good practice to use break. This is why I never mentioned it in class! It's better to write loops that end when the condition is met. Breaks can lead to unexpected code behaviour. While loop example is OK

2/5

PART B

B1 Variables

Logical Stages/Testing - As a general comment I don't think you have consulted the accompanying document "How to lay out Part B of your Assignment"? - it's on Moodle and I also went through it in class. You've written your logical steps more like a narrative. However, some good points are made. No testing shown?

1.5/5

Code - Generally good, you might also want to validate for pay below 0 or the length of the NI number. UserPay should also be a while rather than an if (because they might enter it wrongly more than once).

7/10

B2 Shape

Logical Stages/Testing - Same comment as above for logical stages. Some testing, but only the "happy path" and no screenshots provided.

2.5/5

Code - A reasonable answer. You need to validate the inputs of square and radius to prevent them being negative or wildly out of range. The algorithm for the relative sizes of the square and circle is a bit lacking - comparing the areas will not stop all sizes of square exceeding the circle. It's a better approach to compare squareSide and circleArea.

9/15

B3 Lists

Logical Stages/Testing - Same comment as above for logical stages. Some testing, but again, only the "happy path"

5/10

Code - I'm somewhat confused what you have done with the functions, a lot of it is quite unnecessary and makes your code inefficient. You'd be better off with the lists, one with the items and one with the prices. Nevertheless, a good attempt, but you need to validate the inputs on cost so that they're not negative.

13/20

B4 Turtle

Code - Good use of loop for the Os. You could also have recognised the rotational symmetry on the Z and used a loop there too. Good effort overall.

16/20

General comments:

Nice to see a suitable title page and contents page!
Good layout.

You should not use "I" or "you" or "the author" etc in an academic assignment. This is what we call the active voice, you need to use the passive voice. For example "Memory where I can store, retrieve and change data" write instead -> "Memory where data can be stored retrieved and changed"

OVERALL TOTAL FOR MODULE: 62%

PAGE 3

PAGE 4

PAGE 5

PAGE 6

PAGE 7

PAGE 8

PAGE 9

PAGE 10

PAGE 11

PAGE 12

PAGE 13
