Make sure to do the assignments for BOTH Section 1 and Section 2 below

Section 1: Python Programs

Week 1

1. Copy all the programs from the Weekly Sessions and get them to work

2. Modify any one of the above programs to change its output

3. Program 1: Write a program to display the message “Hello everyone” **three times**, on separate lines on the screen using **3 print()** statements

   Hello everyone
   Hello everyone
   Hello everyone

4. Modify Program 1 to produce the same output BUT using **only 1** print()

   Hint: When you put ‘\n’ in a string, the print() function starts a newline e.g.

   print(‘Bye \n Bye\n’)

outputs

Bye
Bye

5. Write a program that prints out your name, followed by a blank line, followed by your address, followed by a blank line, followed by your telephone number (you may make up an address and a telephone number). Save this program as p1_5.py.

Week 2

1. Create and run the beep program below

```
# beep.py: Just for fun - beep 3 times !!


print("\a \a \a")
```

(a) Modify Program 1 from Week 1, number 3 above, to beep after it displays each line
(b) Modify the program to beep twice before it displays each message

2. Write a program that uses a single print command with a number of arguments to print to the screen the strings "Hello," and "world." The output should include a space between the comma and the word "world".
   Save your program as p2.py.

3. Write a program that uses a single print command with a single argument to print to the screen the concatenation of the strings "Hello," and "world." Again, the output should include a space between the comma and the word "world".
   Save your program as p3.py.

4. Write a program that assigns to a variable the concatenation of the strings "Hello," and "world." and includes a space between the comma and the word "world". The program should then print out the value of this variable.

   Save your program as p4.py.

   For each of the following programs, use assignment statements to give values to variables.

5. Write a program that takes an amount of currency (a float) and an exchange rate to another currency (a float) and prints out the value of the original amount in the other currency. (Use today's exchange rate for two currencies of your choice.)

   Save this program as p5.py.

6. Write a program that takes a single length (a float) and calculates the following:
   - The area of a square with side of that length. (length * length)
   - The volume of a cube with side of that length. (length ** 3)
   - The area of a circle with diameter of that length (3.14 * (length/2)**2))

   You can use 3.1415927 for the value pi.

   Save this program as p6.py.

7. Write a program that takes an amount (a float), and calculates the tax due according to a tax rate of 20%
   Save this program as p7.py.

Week 3

1. Create and runs the programs convert5.py and convert6.py from the section on *Conditional Statements*, in the *Handbook Introduction to Programming*

2. Modify the calculator program calc3.py from the section on *Conditional Statements*, in the *Handbook Introduction to Programming* to handle multiplication and division as well as addition and subtraction.

3. Write a program to read three numbers and display the largest and smallest of the numbers entered.

4. Write a program to simulate a cash register for a single purchase. The program should read the unit cost (real number) of an item and the numbers of items purchased. The program should display the total cost for the items. If the unit cost is greater than 10000, the program should display an error message, "Invalid unit cost – too large". If the unit cost is 0 it should display an error message, "Unit cost cannot be 0".

   Enter unit cost: 5.5
   Enter number of units: 10
   Total cost: 55.0

   Enter unit cost: 0
   Unit cost cannot be 0

5. Write a program to show a menu of areas to be calculated and to calculate the rea chosen by the user. The menu you are to display, is shown in italics below

   Choose the area you wish to calculate from the menu below

   a for the area of a square
   b for the area of a circle
   c for the area of a rectangle

   Enter your choice:

The program should prompt for the dimensions of the area:
   length of a side in the case of a (area = length **2);
   radius in the case of b (area = 3.14 * r**2)
   length and breadth in the case of c (area = length * breadth).

6. Write a program to read a string from the user and display an appropriate message depending on the first character of the string indicating if the character was an uppercase letter, a lowercase letter or a digit ('0' to '9'). Sample outputs are shown below:

   Enter a string: A
   You entered an uppercase letter

   Enter a string: b
   You entered a lowercase letter

   Enter a string: 5

You entered a digit

Enter a string: &
You did not enter a letter or a digit.
You entered &

Week 4

1. Modify Program 4 from Week 3 to allow the user keep entering unit cost and number of units. The program terminates when the user enters -1 for unit cost.

2. Modify Program 5 from Week 3 to use a loop to allow the user keep choosing an area to calculate until the user enters q or Q to quit.

3. Write a program to convert Khat to dollars. The program should continue running until the user enters q or Q to quit e.g.

   Enter Khat or q to quit:

4. Write a program to using a loop to
   read number of hours worked per week (maximum is 100)
   read rate per hour (max is 50)
   compute gross pay (pay before tax)
   compute tax at 10% of pay if pay > 100
   tax is 0 if pay <100
   display result as

   Gross Pay: 200 rate: 10 Hours worked: 40

   Tax: 20

   Net Pay: 180

   Press Y to continue for another employee:

5. Write a program to display 10 lines with
   9 Spaces followed by 1 star on line 1;
   8 spaces followed by 2 stars on line 2,
   7 spaces followed by 3 stars on line 3 and so on

   0 spaces and 10 stars on line 10
   The output should appear as follows:

*

```
  **
 ***
****

*********
```

6. Modify program 5 above to output a what looks like a "tree" as follows

```
  *
 ***
*****
*******

*******************
```

7. Write a program to read a string and display it, replacing any occurrence of A in the string by o:

   Enter a string with an A in it: ABBA
   Replacing A with o gives: oBBo

8. Write a program to read a string and count the number of digits (0,1,2,..9) in it

   Enter a string: 12abc34def56
   There are 6 digits in 12abc34def56

   Hint: if s[i] >='0' **and** s[i] <= '9' then s[i] contains a digit.

9. Write a program to read a string and count the number of uppercase letters (A..Z) and lowercase letters (a..z) in the string

   Enter a string : ABC defg

   3 uppercase letters and 4 lowercase letters in ABC defg

   Hint: if s[i] >='A' **and** s[i] <= 'Z' then s[i] contains an uppercase letter.

Week 5

1. Copy and run the programs create.py, read.py, wc.py and copy.py, from the Handbook.

2. Write a program to create a file called **hours.txt** with the following lines(name and hours worked on separate lines):
   Joe
   34
   Mary
   42
   Jack
   63
   Ann
   21

3. Write a program to read the names and hours worked stored in **hours.txt**. The program should calculate and display the pay for each person assuming a rate of 10 per hour:
   Joe 340
   Mary 420
   Jack 630
   Ann 210

4. Write a program **secret.py** to read a secret word from the user and store it in the file **secret.txt**.

5. Write a program **guess.py** to give the user 4 chances to guess the secret word stored in **secret.txt**.

6. Modify secret.py to allow the user to store as many secret words as they wish, each on a separate line in the file.

7. Modify guess.py to allow the user keep playing the guessing game until (a) they have used all the secret words that were stored in secret.txt or (b) the user decides to quit by entering **quit** as a guess.

8. Write a program that reads a python program, file and checks that brackets (and) are balanced, ie there should never be a situation where there are more right brackets than there are corresponding left brackets and the total number of right brackets should equal the number of left brackets.

   Your program should return the total number of each bracket and a message indicating whether or not the file has balanced brackets.

9. Write a program **display.py** to display a file specified by the user:

   Enter name of file to be displayed: names.txt

   Contents of names.txt are shown here

10. Modify display.py to display the file contents **5 lines at a time**. The program pauses after displaying 5 lines until the user presses either Q to quit or Return to display the next 5 lines. As in previous programs, we read the filename from user and open it appropriately. We then process the file:

```
read line from file
linecount = 1
while ( (not end of file) and (user not finished)
        display line
        linecount = linecount + 1

        if linecount == 5
                linecount = 1
            prompt user to continue displaying or quit
        read next line from file                    # end while loop
```

11. Write a program to **compare** two files specified by the user, displaying a message indicating whether the files are identical or different. This is the basis of a **compare** command provided by most operating systems. The **pseudo code** below outlines an algorithm for this program – convert it to Python. The term pseudo code is used for language that looks like a programming language.

```
prompt user for name of first file say fileA
check if file exists, quitting if it does not
prompt user for name of second file say fileB
check if file exists, quitting if it does not
open fileA
open fileB
read string sa from fileA
read string sb from fileB
while ( (sa == sb) and (not EOF file A) and (not EOF file B) )
            read string sa from fileA
            read string sb from fileB       # end of while loop
if (sa == sb)
            display “Files identical”
else
```

```
            display “Files differ”
```

Note: There are 3 reasons for the while loop to finish:

1. The strings are the same but we have reached the end of the 2 files and hence the file must be the same
2. The strings differ and hence the files differ
3. We have reached the end of 1 file but not the other file and hence the file differ

Section 2: Weekly Exercises

Week 1

1. Make the following deliberate errors in a Python program and explain what happens
    a. Omit ” at end of a string
    b. Omit “(“ from a print statement
    c. Omit “)” from a print statement
    d. Misspell print as “prince”

1. Explain the concept of **variable**
2. Explain what is meant by the ***type*** of variable

3. Why do we need to use meaningful variable names ?
4. Which of the following are valid Python variable names

    a. Tax1123
    b. Tax456
    c. 12tax
    d. Tax_code
    e. Tax-code
    f. Tax.code
    g. print

Week 2

1. What is the difference between an algorithm and a program ?
2. What is a syntax error ?
3. Why should you use comments in a program ?
4. What type does the `input()` function return ?
5. How can we convert the type returned by `input()` to a `float` ?

Week 3

1. What will the following code fragments display:

   (a)

```
n = 10

if n == 10:
        print("n is 10")
else:
        print("n is not 10")
```

   (b)

```
n = 0

if n =10
        print("n is 10")
else
        print("n is not 10")
```

   Explain the error in this code fragment

2. What is a Boolean expression ?

3. What values can a Boolean expression take ?

Week 4

2. What sequences will the following range functions produce
   a. range(10)
   b. range(10, 14)
   c. range(2,10)
   d. range(10, 4, -1)
   e. range(100, 10, -10)
   f. range(1, 10)
   g. range (10, -1,-1)
   h. range(5, -2, -1)

3. Will the following loop finish ?

```
j = 0
while j < 10:
        print("j = ", j)
j = j + 1
```

Week 5

1. What is the difference between opening a file for reading and for writing ?
2. Why should we check if a file exists before we open it for reading ?
3. Why do we need to close a file when we are finished with it ?
4. What function do we use to terminate a Python program? Why is this useful?