UniSA STEM

COMP 1039 Problem Solving and Programming

Practical 3

While loops and Strings

Please make sure you have completed Practical 2 before continuing with Practical 3.

1. Properly indent the following statements (you may refer to the examples in the weekly slides if you need help with this question).

   a)

```
a=1  b=1  c=1
if a == b: print('In if statement.')  print('Equal.')  else:
print('In else statement.')  print('Not equal.')
```

   The above code should produce the following output:

```
In if statement.
Equal.
```

   b)

```
max = 5
count = 0  k = 0
while k < max:
count = count + 1
print('In while loop.')
k = k + 1
print('Count is: ', count)
```

   The above code should produce the following output:

```
In while loop.
In while loop.
In while loop.
In while loop.
In while loop.
Count is:  5
```

2. What is the output produced by the following code? Check your answers using Python.

a)

```
k = 1
while k <= 5:
    print('Stuck in a loop!')
    k = k+1
else:
    print('okay then...')
```

b)

```
k = 6
while k >= 2:
    print('Still looping.')
    k = k - 2
```

c)

```
num1 = 2
num2 = 4
if (num1 < num2):
    print('Yes.')
```

d)

```
str1 = "kramer"
str2 = "george"
if str1 == str2:
    print("Equal.")
else:
    print("Not equal.")
```

e)

```
mark = 78
if mark >= 85:
    print('HD')
elif mark >= 75:
    print('D')
elif mark >= 65:
    print('C')
elif mark >= 55:
    print('P1')
elif mark >= 50:
    print('P2')
elif mark >= 40:
    print('F1')
else:
    print('F2')
```

3. What is the output produced by the following programs (a-f)?

a)

```
str1 = 'Wednesday Thursday Friday'
new_string = ''

index = 0
while index < len(str1):
    if str1[index].isupper():
        new_string = new_string + str1[index]

    index = index + 1

new_string = new_string + '!?!'
print(new_string)
```

b)

```
str1 = 'Wednesday Thursday Friday'
new_string = ''

index = 0
while index < len(str1):
    if str1[index].isupper():
        new_string = str1[index] + new_string

    index = index + 1

new_string = new_string + '!?!'
print(new_string)
```

c)

```
str1 = '348924731'
new_string = ''

index = 0
while index < len(str1):
    if index % 2 == 0:
        new_string = new_string + str(str1[index])

    index = index + 1

print(new_string)
```

d)

```
str1 = '348924731'
result = 0

index = 0
while index < len(str1):
    if index % 2 == 0:
        result = result + int(str1[index])

    index = index + 1

print(result)
```

e)

```
index = 0
result = 0

while index < 10:

    result = result + index

    index = index + 1

print('Result is: ', index, result)
```

f)

```
index = 0
result = ''

while index < 10:

    result = result + str(index)

    index = index + 1

print('Result is: ', index, result)
```

4. Create a file in order to complete this exercise. Many games require a player to roll two dice. The number on each die can vary from 1 to 6.
    a. Use the `random.randint()` function to create a simulation of one roll of one die. Store the result in a variable called `die1`. (hint: remember to place `import random` at the top of your file so that you may make use of the randint() function).
    b. Repeat part a. to create a simulation of the value rolled with a second die. Store the result in a variable called `die2`.
    c. Display the results of the die to the screen in the following format:

       If the values of the die are the same – display the message you rolled a pair of… i.e.:

       ```
       You rolled a pair of 2s!
       ```

       Otherwise, display the values of the die. i.e.:

       ```
       You rolled a 2 and a 6
       ```

5. Create a file in order to complete this exercise. Write a program that prompts the user to enter a number within the range of 1 through 10. The program should display the Roman numeral version of that number. If the number is outside the range of 1 through 10, the program should display an error message. The following table shows the Roman numerals for the numbers 1 through 10.
(Modified: Gaddis, Tony. Programming Exercises, Chapter 4).

Number	Roman Numeral
1	I
2	II
3	III
4	IV
5	V
6	VI
7	VII
8	VII
9	IX
10	X

Your output should be presented as follows (user input appears in bold):

```
Enter an integer from 1-10: 7
VII
```

6. Create a file in order to complete this exercise. The date 10 June 1960, is special because when it is written in the following format, the day times the month equals the year:

10/6/60

Write a program that asks the user to enter a day, a month (in numeric form), and a two-digit year. The program should then determine whether the day times the month equals the year. If so, it should display a message saying the date is magic. Otherwise, it should display a message saying the date is not magic.
(Modified: Gaddis, Tony. Programming Exercises, Chapter 4).

Your output should be presented as follows (user input appears in bold):

```
Enter the day of month: 10
Enter the month in numeric form: 6
Enter the year in two digit format: 60

The date 10/6/60 is a magic date.
```

7. Create a file in order to complete this exercise. Write a program that generates a random number between 1-10, asks the user to guess the number and displays '`Well done- you guessed it!`' if the user guesses the number correctly, otherwise displays the message '`Too bad – better luck next time!`' if the user guesses incorrectly.

   Your output should be presented as follows (user input appears in bold to differentiate it from text that is displayed to the screen by the program):

   Example 1:
   ```
   Please enter your guess: 8

   Too bad - better luck next time!
   ```

   Example 2:
   ```
   Please enter your guess: 4

   Well done - you guessed it!
   ```

   For testing purposes, you may wish to use a print statement in order to display the random number generated (that way you will know whether your program is working or not). Once you are sure your program is working correctly, you may remove the print statement.

Checkpoint: Please make sure your supervisor sees both your program output and code.

Question 8 (part a)

Create a file in order to complete this exercise. Write a program that generates 100 random numbers (in the range of 1-1000) and keeps a count of how many of those random numbers are even and how many are odd. Display the results to the screen as seen in the sample output below. Hint: Use a while loop to loop 100 times. (Modified: Gaddis, Tony. Programming Exercises, Chapter 6).

Your output should be presented as follows:

Sample output 1:
```
Out of 100 random numbers, 56 were odd, and 44 were even.
```

Sample output 2:
```
Out of 100 random numbers, 39 were odd, and 61 were even.
```

Question 8 (part b)

Add another while loop which repeats part a 10 times. Display the results to the screen as seen in the sample output below.

Sample output 1:

```
Out of 100 random numbers, 56 were odd, and 44 were even.
Out of 100 random numbers, 60 were odd, and 40 were even.
Out of 100 random numbers, 47 were odd, and 53 were even.
Out of 100 random numbers, 54 were odd, and 46 were even.
Out of 100 random numbers, 48 were odd, and 52 were even.
Out of 100 random numbers, 53 were odd, and 47 were even.
Out of 100 random numbers, 46 were odd, and 54 were even.
Out of 100 random numbers, 52 were odd, and 48 were even.
Out of 100 random numbers, 53 were odd, and 47 were even.
Out of 100 random numbers, 48 were odd, and 52 were even.
```

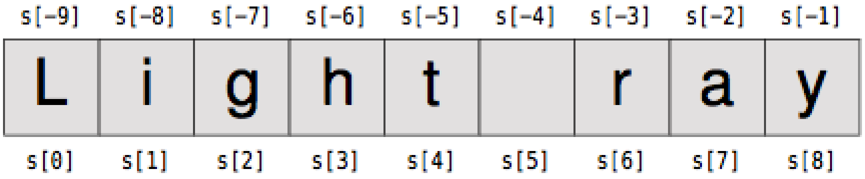
Sample output 2:

```
Out of 100 random numbers, 49 were odd, and 51 were even.
Out of 100 random numbers, 55 were odd, and 45 were even.
Out of 100 random numbers, 55 were odd, and 45 were even.
Out of 100 random numbers, 50 were odd, and 50 were even.
Out of 100 random numbers, 51 were odd, and 49 were even.
Out of 100 random numbers, 56 were odd, and 44 were even.
Out of 100 random numbers, 39 were odd, and 61 were even.
Out of 100 random numbers, 43 were odd, and 57 were even.
Out of 100 random numbers, 47 were odd, and 53 were even.
Out of 100 random numbers, 51 were odd, and 49 were even.
```

Checkpoint: Please make sure your supervisor sees both your program output and code.

Question 9

The string `s = "Light ray"` has the indices as follows:



s[-9]	s[-8]	s[-7]	s[-6]	s[-5]	s[-4]	s[-3]	s[-2]	s[-1]
L	i	g	h	t		r	a	y
s[0]	s[1]	s[2]	s[3]	s[4]	s[5]	s[6]	s[7]	s[8]

We can access values using:

s[start]	Value at index start Example: s[3] has value 'h'
s[start:end]	Slice from start to end-1 Examples: s[1:5] is "ight" s[:5] is "Light" s[6:] is "ray"
s[start:end:step]	From start to end-1 with step size Examples: s[::2] is "Lgtry" s[::-2] is "yrtgL" s[::-1] is "yar thgiL" s[0:5:2] is "Lgt"

What is the output produced by the following code? Check your answers using Python.

```
name1 = 'Homer'
name2 = 'Simpson'
name3 = 'Donuts'

print(name1 + ' ' + name2)

print(name1 + '\'s ' + name3)

print('m'*6 + ' ' + name3)

print(name1[2])

print(name3[2:6])

print(name1[0:4:1])

print(name2[-2])

print(len(name2))

fullName = name1 + ' ' + name2

if fullName == 'Homer Simpson':
    print('Doh!')
else:
    print('Bart - is that you?')
```

Question 10

A few more string methods:

s.capitalize()	Returns a copy of s with first character capitalized and the rest lowercase.
s.center(width, char)	Returns a copy of the string s centred in a string of length width using char for padding. Default padding is space. Also s.ljust() and s.rjust()
s.count(t, start, end)	Count occurrences of string t within s[start:end] - start, end are optional.
s.isalnum()	Returns True if s is non-empty and every character alphanumeric
s.isalpha()	Returns True if s is non-empty and every character is alphabetic
s.isdigit()	Returns True if s is non-empty and every character is a digit.
s.islower()	Returns True if s is non-empty, and all cased characters are lowercase. Also s.isupper()
s.lower()	Returns a copy of the string converted to lower case. Also s.upper()
s.replace(t, u, n)	Replaces within s occurrences of t by u (up to n times if n given).
s.split(sep, maxsplit)	Returns a list of words in the string, using sep as the delimiter string. If maxsplit is given, at most maxsplit splits are done. (We will cover lists shortly).
s.startswith(x, start, end)	Returns true if s[start:end] starts with the string x. There is an endswith() method as well.
s.strip(chars)	Returns a copy of s with leading and trailing white spaces (or the characters in chars) removed.
s.swapcase()	Returns a copy of the string with the case of characters swapped.

For more string methods (refer to online docs): http://docs.python.org/library/stdtypes.html#string-methods

Given the following string:

```
str1 = "These pretzels are making me thirsty!"
```

Write code to:

1. Count how many times the letter 'e' appears in the string.
2. Determine whether every character in the string is a digit.
3. Convert the string to lowercase.
4. Convert the string to uppercase.
5. Replace all white space characters with character '-'.

Question 11

Create a file in order to complete this exercise. Write a program that takes in two integers as input from the user. Compute the difference between their product and their sum and display the result to the screen.

Question 12

Describe what the following program does. Check your answer using Python.

```
x = int(input("Enter a number: "))
display = x * "-"
print(display)
```

Please make sure you save and keep all of your practical and assignment work. Please ask your supervisor if you are having difficulties doing so.

End of practical 3.