Government of Western Australia
North Metropolitan TAFE

Assessment Task: Lab 4

Qualification national code and title	22603VIC Certificate IV in Cybersecurity
Unit/s national code/s and title/s	**ICTPRG434 - Automate processes** **ICTPRG435 - Write script for software applications**

Assessment type (☑):

☐ Questioning (Oral/Written)
☐ Practical Demonstration
☐ 3rd Party Report
☒ Other – Lab

Assessment Resources:

The base requirements this assessment task include:

- IDE or editor for developing Python programs *(only IDLE and PyCharm supported by the college)*
- Access to Office 365 & Microsoft Word
- Virtual machine

You may not need all these for every part in this assessment

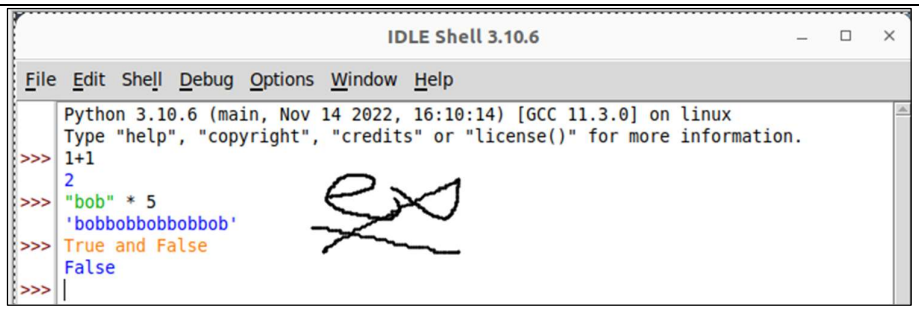
Assessment Due:

This assessment is due after the weekly session, **Week 4, Friday 17:00**.

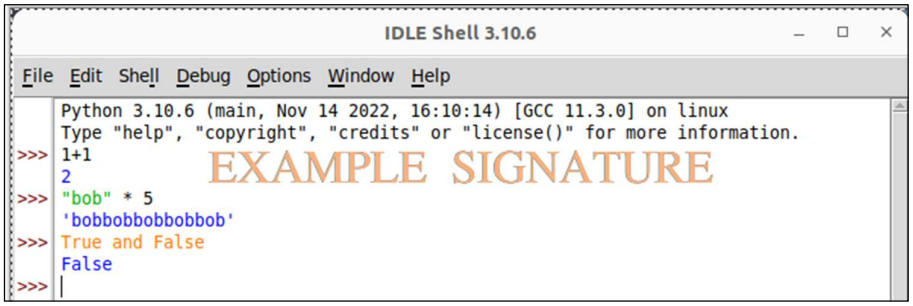
Assessment Instructions:

1. Your code must be written in IDLE or PyCharm IDEs. If you are using a different IDEs or a different structure for your application, then assistance from your lecturers may be limited (at best). Discuss with your lecturer before straying too far off the path!
2. All resources used should be referenced with the question. Answers may not be copied and pasted from any resource. All answers must be reworded to display your understanding.
3. You may only use Python functionality, methods and libraries which were taught in this unit.
4. First line of code in a program should have the student's name and number, as proof of authenticity.
5. Screenshots of all programs must be included **in** this document, with the appropriate question.
6. Screenshots of testing, showing your code works as intended, should be included with the relevant question.
7. Python programs should be named: XXX_Lab##_SYY_QZZ
   Replace XXX with your initials
   Replace ## with Lab number
   Replace YY with Section number,
   Replace ZZ with Question number
8. It is a submission requirement that all screen shots be signed in some way. Some acceptable examples of signed screen shots are shown below.

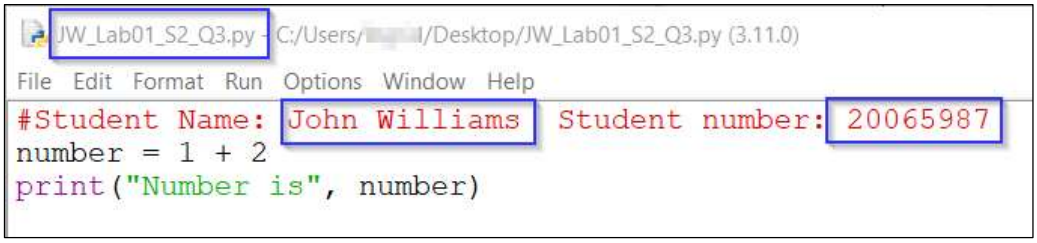
Qualification national code and title	22603VIC Certificate IV in Cybersecurity
Unit/s national code/s and title/s	**ICTPRG434 - Automate processes** **ICTPRG435 - Write script for software applications**



Example 1: Signed using a simple drawing tool.



Example 2: Water marked signature.



Example 3: Program named as prescribed, as well as first line comment with student name and number. Program saved as pre-described.

9. All python programs must be included in the submission, as well as this document.

Assessment Instrument:

Government of Western Australia
North Metropolitan TAFE

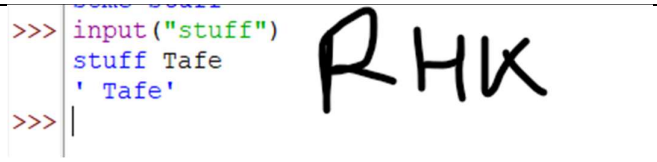
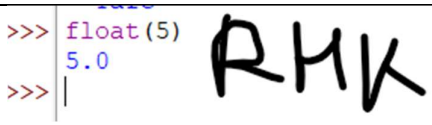
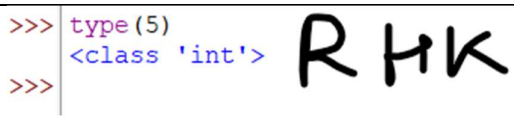
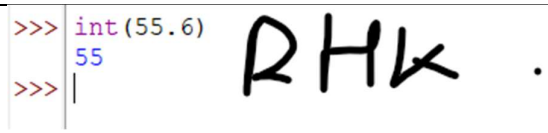
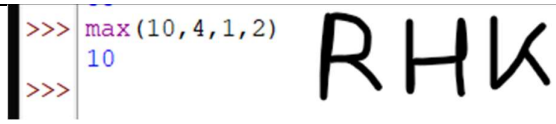
Assessment Task: Lab 4

Qualification national code and title	22603VIC Certificate IV in Cybersecurity
Unit/s national code/s and title/s	**ICTPRG434 - Automate processes** **ICTPRG435 - Write script for software applications**

Section 1: Built in functions

Begin this lab by exploring some built in functions. Run the following exercises in the Python shell/interpreter.

1. Run the following operations and evaluate the output:

Code	**Screenshot**
print("some stuff")	
input("stuff")	
float(5)	
type(5)	
int(55.6)	
max(10,4,1,2)	

2. Provide the name and description of at least one more function built into Python.

An object can be converted into its string representation using Python's built-in str() function.

3. What does the **min()** function do?

One of Python's built-in functions for figuring out the smallest object is min().

Qualification national code and title	22603VIC Certificate IV in Cybersecurity
Unit/s national code/s and title/s	**ICTPRG434 - Automate processes** **ICTPRG435 - Write script for software applications**

Section 2: Type conversion functions

Use Python shell/interpreter and built-in functions to convert the following data to a different type.

1. Convert the **integer 32** to a **float**

```
>>> float(32)
32.0
>>>
```

RHK

2. Convert the **string "7"** to an **integer**.

```
>>> int("7")
7
>>>
```

RHK

3. Convert the **integer 1** to a **Boolean**.

```
>>> bool(1)
True
>>>
```

RHK

4. Convert the **float 98.8** to an **integer**

```
>>> int(98.8)
98
>>>
```

RHK

5. Why does **98.8** converted to an **integer** return as **98** and not **99** as would be expected if rounding?

The decimal portion of a float is only truncated, not rounded, when it is converted to an int in Python using the int() method.

Section 3: Other useful built-in functions

There are other more complex built-in functions in Python ready to be used. Try out some of the following functions in the Python shell/interpreter

1. Run the following operations for the random library and evaluate the output

Code	**Explanation / Comment**	**Screenshot**

import random	This import code will add the	`>>> import random`
		`>>> random.randint(1,5)`
		`1`
		RHK

Qualification national code and title	22603VIC Certificate IV in Cybersecurity
Unit/s national code/s and title/s	**ICTPRG434 - Automate processes** **ICTPRG435 - Write script for software applications**

	"random" library to your shell.	

random.randint(1,5)	Run this code three times.	`>>> random.randint(1,5)`
		`2`
		`>>>`
		RHK

random.random()	Run this code three times.	`>>> random.random()`
		`0.9943253618364432`
		`>>>`
		RHK

random.randrange(50)	Run this code three times.	`>>> random.randrange(50)`
		`12`
		`>>>`
		RHK

2. The random library can be combined with some other techniques to make some interesting results. Type the following lines into the interpreter and screen shot the results:

```
import random
deck = "king queen jack 10 9 8 7 6 5 4 3 2 1".split()
print(deck)
random.shuffle(deck)
print(deck)
```

```
>>> deck = "king queen jack 10 9 8 7 6 5 4 3 2 1".split()
>>> print(deck)
['king', 'queen', 'jack', '10', '9', '8', '7', '6', '5', '4', '3', '2', '1']
>>> random.shuffle(deck)
>>> print(deck)
['4', 'queen', 'jack', '9', '7', '10', 'king', '1', '6', '3', '2', '5', '8']
>>>
```

RHK

Ln: 39 Col: 0

3. Type the following lines into the interpreter and screen shot the results:

```
import random
result = "win lose draw".split()
random.choice(result)
```

```
>>> result = "win lose draw".split()
>>> random.choice(result)
'lose'
>>>
```

RHK

Qualification national code and title	22603VIC Certificate IV in Cybersecurity
Unit/s national code/s and title/s	**ICTPRG434 - Automate processes** **ICTPRG435 - Write script for software applications**

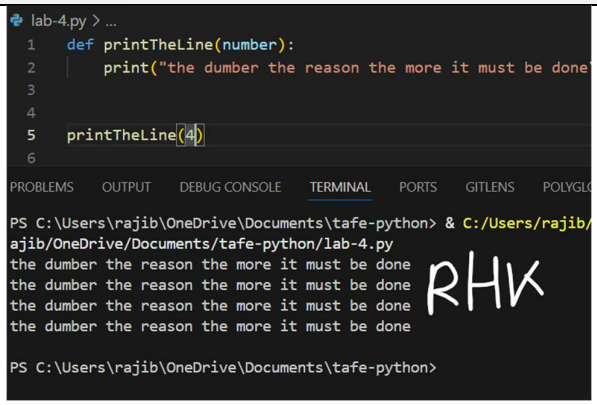
Section 4: Building functions

You will frequently be required to create your own functions with parameters and arguments. Using a code file, create the following functions and run them to demonstrate they work:

1. Use the provided code to create and call a function that will print the words "the dumber the reason the more it must be done" as many times as asked using a parsed parameter.

```
def printTheLine(number):
    print("the dumber the reason the more it must be done\n"*number)

printTheLine(4)
```

Code	**Output**
def printTheLine(number): print("the dumber the reason the more it must be done\n"*number) printTheLine(4)	

2. Use the provided code to create and call a function that will pick a number at random for a user to guess!

```
import random

def guessTheNumber(startNumber,endNumber,guess):
    theAnswer = random.randint(startNumber,endNumber)
    print(f"The answer is {theAnswer}")
    if theAnswer == guess:
        return True
    else:
        return False

lowNumber = int(input("Enter the lowest number: "))
highNumber = int(input("Enter the highest number: "))
userGuess = int(input(f"Enter your guess between {lowNumber}-{highNumber}: "))

if guessTheNumber(lowNumber,highNumber,userGuess):
    print("You guessed it!")
else:
    print("Oops, wrong")
```

Code	**Output**

Qualification national code and title	22603VIC Certificate IV in Cybersecurity
Unit/s national code/s and title/s	**ICTPRG434 - Automate processes** **ICTPRG435 - Write script for software applications**

```
import random

def guessTheNumber(startNumber, endNumber, guess):

    theAnswer = random.randint(startNumber, endNumber)

    print(f"The answer is {theAnswer}")

    if theAnswer == guess:

        return True

    else:

        return False

lowNumber = int(input("Enter the lowest number: "))

highNumber = int(input("Enter the highest number: "))

userGuess = int(

    input(f"Enter your guess between {lowNumber} - {highNumber}: "))


if guessTheNumber(lowNumber, highNumber, userGuess):

    print("You guesses it!")

else:

    print("Oops, wrong")
```

3. The last example you made used **f-strings**. What is an **f-string**?

Code	**Output**

Qualification national code and title	22603VIC Certificate IV in Cybersecurity
Unit/s national code/s and title/s	**ICTPRG434 - Automate processes** **ICTPRG435 - Write script for software applications**

f-string, or formatted string literal, which offers a clear and easy-to-understand method of embedding Python expressions inside string literals.

```
lowNumber = int(input("Enter the lowest number: "))
highNumber = int(input("Enter the highest number: "))
userGuess = int(
    input(f"Enter your guess between {lowNumber} - {highNumber}: "))
```

```
PROBLEMS   OUTPUT   DEBUG CONSOLE   TERMINAL   PORTS   GITLENS   POLYGLOT NOTEBOOK

PS C:\Users\rajib\OneDrive\Documents\tafe-python> & C:/Users/rajib/AppData/Local
on311/python.exe c:/Users/rajib/OneDrive/Documents/tafe-python/lab-4.py
Enter the lowest number: 1
Enter the highest number: 10
Enter your guess between 1 - 10: 5
PS C:\Users\rajib\OneDrive\Documents\tafe-python>
```

RHK

4. Create a function that will print “Welcome to Arrestaurant. May I take your statement?” whenever called.

Code

```
def greetings():
    print("Welcome to Arrestaurant. May I take your statement?")

greetings()
```

Output

```
def greetings():
    print("Welcome to Arrestaurant. May I take your statement?")


greetings()

PROBLEMS   OUTPUT   DEBUG CONSOLE   TERMINAL   PORTS   GITLENS   POLYGLOT NOTEBO

PS C:\Users\rajib\OneDrive\Documents\tafe-python> & C:/Users/rajib/AppData/
e/Documents/tafe-python/lab-4.py
Welcome to Arrestaurant. May I take your statement?
PS C:\Users\rajib\OneDrive\Documents\tafe-python>
```

RHK

5. Create a function that will accept an integer and print to the terminal if integer is above or below 10.

Code

```
def compareWith10(number):
    if number > 10:
        print("The number is greater than 10")
    elif number < 10:
        print("The number is less than 10")
    else:
        print("The number is equal to 10")

compareWith10(9)
```

Output

```
def compareWith10(number):
    if number > 10:
        print("The number is greater than 10")
    elif number < 10:
        print("The number is less than 10")
    else:
        print("The number is equal to 10")


compareWith10(9)

PROBLEMS   OUTPUT   DEBUG CONSOLE   TERMINAL   PORTS   GITLENS   POLYGLO

PS C:\Users\rajib\OneDrive\Documents\tafe-python> & C:/Users/rajib/A
e/Documents/tafe-python/lab-4.py
The number is less than 10
PS C:\Users\rajib\OneDrive\Documents\tafe-python>
```

RHK

6. Create a function that will simulate rolling a singular six-sided dice, it should pick a number between 1 and 6 then **return** it to the user

Code

Output

Government of Western Australia
North Metropolitan TAFE

Assessment Task: Lab 4

Qualification national code and title	22603VIC Certificate IV in Cybersecurity
Unit/s national code/s and title/s	**ICTPRG434 - Automate processes** **ICTPRG435 - Write script for software applications**

```
def diceGame():
    return random.randint(1, 6)

print(diceGame())
```