Introduction to Programming

Lab Worksheet

Week 1

Python

Name: Nikita Sah

Level 4 Section: A

British Id: 10011

Level 4 BSc. Hons Computing

Subject: Fundamental Of Computer Programming (FOCP)

The British College (TBC)

Task:

1. Try inputting and executing the code below:

Answer:

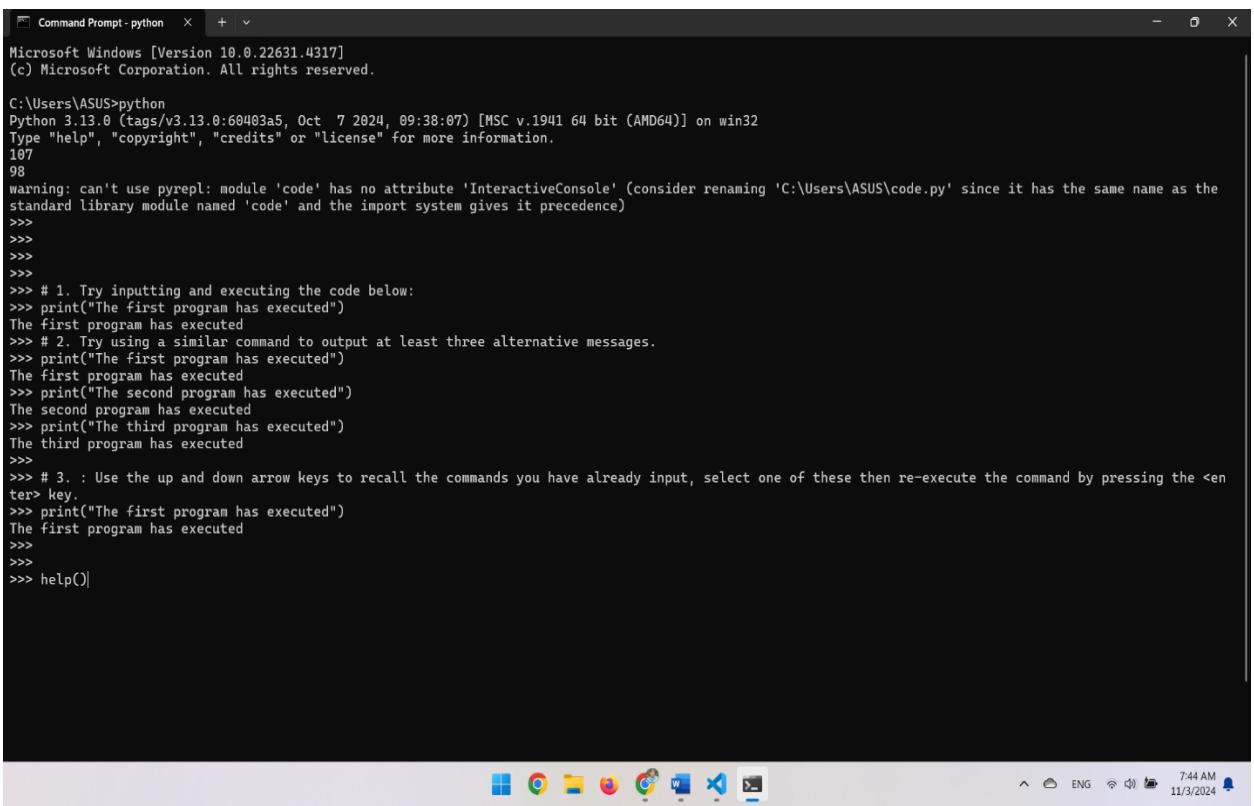
```
print("The first program has executed")
print("_____________________________________")
```

2. Try using a similar command to output at least three alternative messages.

Answer:

```
print("The first program has executed")
print("The second program has executed")
print("The third program has executed")
```

3. Use the up and down arrow keys to recall the commands you have already input, select one of these then re-execute the command by pressing the <enter> key.

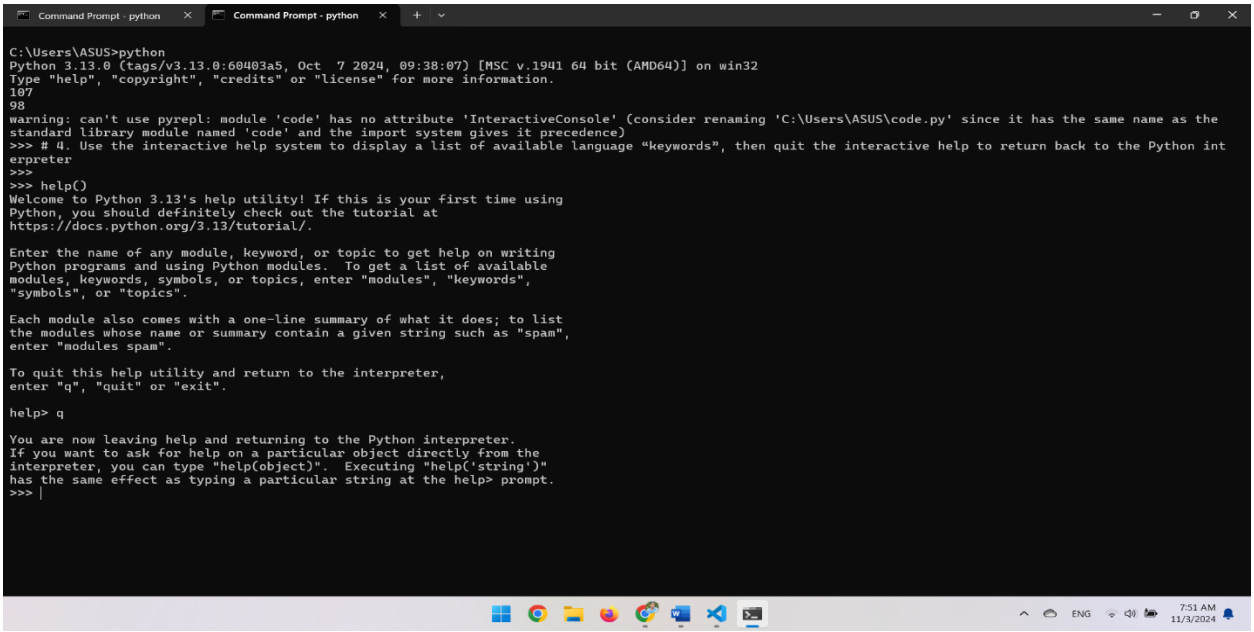


4. Use the interactive help system to display a list of available language “keywords”, then quit the interactive help to return back to the Python interpreter.

Answer:

```
>>>help()

help>q
```



```
C:\Users\ASUS>python
Python 3.13.0 (tags/v3.13.0:60403a5, Oct  7 2024, 09:38:07) [MSC v.1941 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license" for more information.
107
98
warning: can't use pyrepl: module 'code' has no attribute 'InteractiveConsole' (consider renaming 'C:\Users\ASUS\code.py' since it has the same name as the
standard library module named 'code' and the import system gives it precedence)
>>> # 4. Use the interactive help system to display a list of available language “keywords”, then quit the interactive help to return back to the Python int
erpreter
>>>
>>> help()
Welcome to Python 3.13's help utility! If this is your first time using
Python, you should definitely check out the tutorial at
https://docs.python.org/3.13/tutorial/.

Enter the name of any module, keyword, or topic to get help on writing
Python programs and using Python modules.  To get a list of available
modules, keywords, symbols, or topics, enter "modules", "keywords",
"symbols", or "topics".

Each module also comes with a one-line summary of what it does; to list
the modules whose name or summary contain a given string such as "spam",
enter "modules spam".

To quit this help utility and return to the interpreter,
enter "q", "quit" or "exit".

help> q

You are now leaving help and returning to the Python interpreter.
If you want to ask for help on a particular object directly from the
interpreter, you can type "help(object)".  Executing "help('string')"
has the same effect as typing a particular string at the help> prompt.
>>>
```

5. Enter a single ‘help’ command to look up information about the ‘input’ object then return back to the interpreter.

Answer:

```
>>>help()

help>q
```

```
Command Prompt - python    Command Prompt - python
standard library module named 'code' and the import system gives it precedence)
>>>
>>>
>>> # 5. Enter a single 'help' command to look up information about the 'input' object then return back to the interpreter.
>>>
>>> help()
Welcome to Python 3.13's help utility! If this is your first time using
Python, you should definitely check out the tutorial at
https://docs.python.org/3.13/tutorial/.

Enter the name of any module, keyword, or topic to get help on writing
Python programs and using Python modules.  To get a list of available
modules, keywords, symbols, or topics, enter "modules", "keywords",
"symbols", or "topics".

Each module also comes with a one-line summary of what it does; to list
the modules whose name or summary contain a given string such as "spam",
enter "modules spam".

To quit this help utility and return to the interpreter,
enter "q", "quit" or "exit".

help> input
Help on built-in function input in module builtins:

input(prompt='', /)
    Read a string from standard input.  The trailing newline is stripped.

    The prompt string, if given, is printed to standard output without a
    trailing newline before reading input.

    If the user hits EOF (*nix: Ctrl-D, Windows: Ctrl-Z+Return), raise EOFError.
    On *nix systems, readline is used if available.

help> q

You are now leaving help and returning to the Python interpreter.
If you want to ask for help on a particular object directly from the
interpreter, you can type "help(object)".  Executing "help('string')"
has the same effect as typing a particular string at the help> prompt.
>>>
```

6. Try quitting and then restarting the interpreter several times

Answer:

>>>^Z or

>>>exit()

>python

```
Command Prompt - python    Command Prompt - python
    On *nix systems, readline is used if available.

help> q

You are now leaving help and returning to the Python interpreter.
If you want to ask for help on a particular object directly from the
interpreter, you can type "help(object)".  Executing "help('string')"
has the same effect as typing a particular string at the help> prompt.
>>>
>>>
>>>
>>> quit()

C:\Users\ASUS>python
Python 3.13.0 (tags/v3.13.0:60403a5, Oct  7 2024, 09:38:07) [MSC v.1941 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license" for more information.
107
98
warning: can't use pyrepl: module 'code' has no attribute 'InteractiveConsole' (consider renaming 'C:\Users\ASUS\code.py' since it has the same name as the
standard library module named 'code' and the import system gives it precedence)
>>>
>>>
>>> # 6. Try quitting and then restarting the interpreter several times
>>>
>>> ^Z


C:\Users\ASUS>python
Python 3.13.0 (tags/v3.13.0:60403a5, Oct  7 2024, 09:38:07) [MSC v.1941 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license" for more information.
107
98
warning: can't use pyrepl: module 'code' has no attribute 'InteractiveConsole' (consider renaming 'C:\Users\ASUS\code.py' since it has the same name as the
standard library module named 'code' and the import system gives it precedence)
>>> |
```

7. Try inputting and executing the code below:

45 + 20

Answer:

65

8. Input then execute the following expressions (note: you will have to re-enter each expression separately). Ensure you understand each operator and the result produced.

10 + 20 - 15

10 * 5

100 / 33

100 // 33

10 ** 2

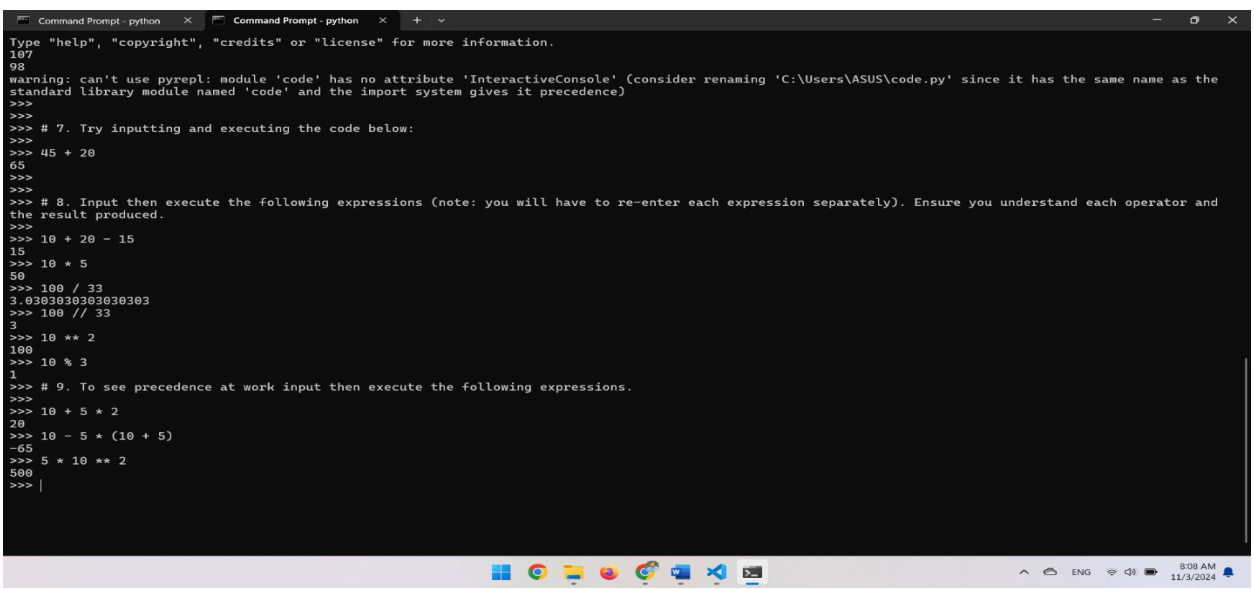
10 % 3

9. To see precedence at work input then execute the following expressions.

10 + 5 * 2

10 - 5 * (10 + 5)

5 * 10 ** 2



10. Input and execute the following expressions, then compare the results to those of the previous task.

(10 + 5) * 2

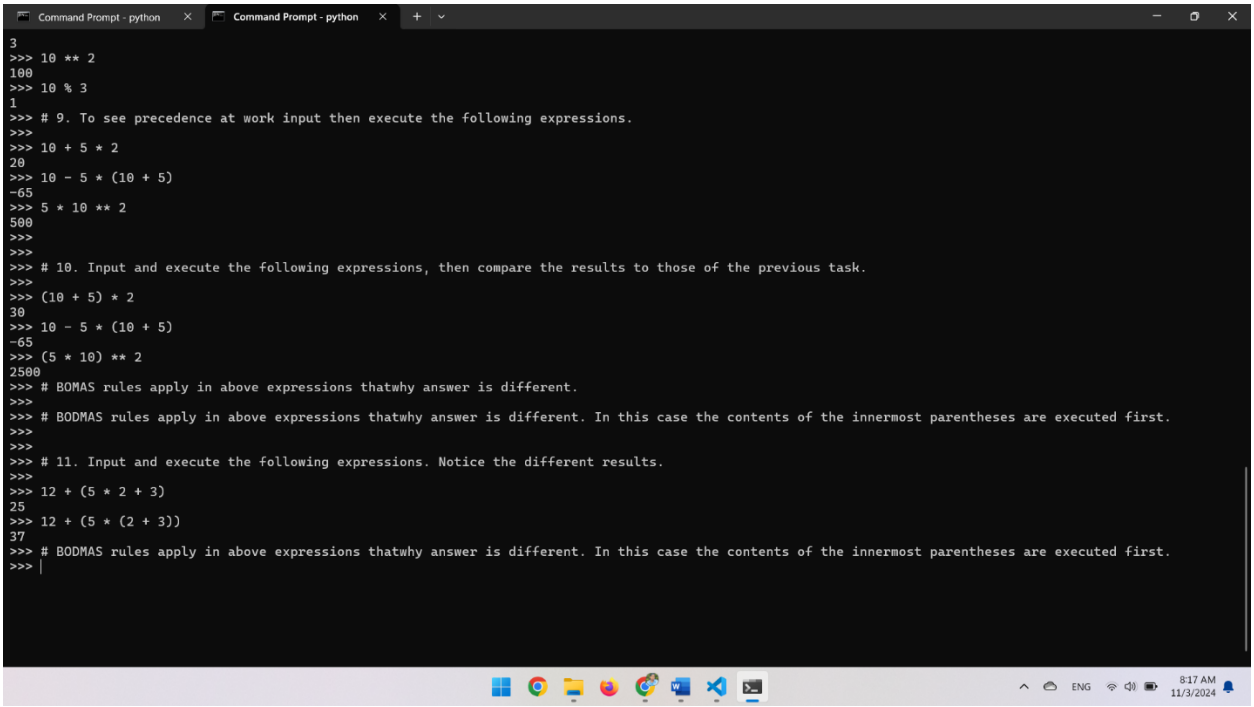
10 - 5 * (10 + 5)

(5 * 10)** 2

11. Input and execute the following expressions. Notice the different results.

12 + (5 * 2 + 3)

12 + (5 * (2 + 3))



12. Look at each of the phrases below and ensure you understand what each of these means. For any that you do not understand, do a little research to find a definition of

each term. This research may involve looking back over these notes, or the associated lecture notes. It may also involve searching for these terms on the Internet.

- Source code
- Machine code
- Interpreter
- Compiler
- 2GL, 3GL, 4GL
- Executable

- Expressions
- Operators and Operands
- Syntax Errors
- Logical Errors

Answer:

Source code: Source code is the set of instructions that a programmer writes to create software.

Machine code: Machine code (also known as machine language or native code) is a low level programming language in the form of hexadecimal or binary instructions that execute instructions directly on the computers' CPU.

Interpreter: An interpreter is a program that directly executes the instructions without compilation of the written program code.

Compiler: A compiler is a computer program that translates source code written in a high-level programming language into machine code, bytecode, or another programming language.

2GL, 3GL, 4GL: 2GL stands for second-generation programming language, a low-level programming language that uses assembly language.

3GL stands for "Third Generation Language" and is a high-level programming language that's more programmer-friendly and machine-independent.

4GL is a scripting programming language that is interpreted during runtime, used in querying the database or in server.

Executable: An executable is a file that contains a program that can be run by a computer's operating system or a software application.

Expressions: An expression is a combination of variables, operators, literals, and function calls that evaluates to a value.

Operators and Operands: An operator is a symbol that tells the compiler or interpreter to perform special mathematical or logical functions.
e.g. a= b + c

An operand is a value or expression that is used to perform an operation.

Syntax Errors: A syntax error is an error in the syntax of a coding or programming language, entered by a programmer.

Logical Errors: A logical error is an error which occurs when the program compiles and runs without any syntax, run-time, or linker errors, but the output is incorrect or unexpected.