



Islington college
(इरिलिङ्टन कलेज)

Module Code & Module Title

CS4051NI Fundamentals of Computing

Assessment Weightage & Type

60% Individual Coursework

Year and Semester

2019-20 Autumn

Student Name: Ishan Gurung

Group: N7

London Met ID:19031315

College ID: NP01NT4A190139

Assignment Due Date:9th June,2020

Assignment Submission Date:9th June 2020

I confirm that I understand my coursework needs to be submitted online via Google Classroom under the relevant module page before the deadline in order for my assignment to be accepted and marked. I am fully aware that late submissions will be treated as non-submission and a mark of zero will be awarded.

Contents

Introduction.....	5
Model:	6
1-bit adder circuit:	6
8-bit parallel circuit.....	8
Algorithm:	8
numValidation model:.....	8
binaryConversion model	8
Addition model:	9
Reverse model.....	9
Main model.....	10
Pseudocode	10
Main program module:	10
Flowchart:.....	14
Data Structure:.....	17
Testing	18
Test no 1.....	18
Test no 2:.....	20
Test no 3.....	22
Test no 4.....	24
Test no 5.....	25
Conclusion.....	28

Figure 1 Full adder circuit	6
Figure 2 8- bit parallel circuit	8
Figure 3 Flowchart.....	14
Figure 4 input value	19
Figure 5 result of test 1	20
Figure 6 test no 2.....	21
Figure 7 input value	21
Figure 8 result of test no 2.....	22
Figure 9 input value	23
Figure 10 result of test no 3	23
Figure 11 input value of test 4	24
Figure 12 result of test 4.....	25
Figure 13 input value of test 5	26
Figure 14 result of test 5.....	27

Table 1 Truth table	7
Table 2 test 1.....	18
Table 3 test no 3.....	22
Table 4 test no 4.....	24
Table 5 test no 5.....	26

Introduction

On date 5th April, 2020 coursework was released. So, there is the software python 3.8 IDLE (Integrated and learning Environment) to do coursework's coding part. It is the Integrated and learning Environment for python. The Python installer for Windows contains the Idle module by default. IDLE is not default in python for Linux. In Linux it needs to be installed using the respective package managers. Just like Python Shell, IDLE can be execute a single statement and also create, modify, and execute Python Scripts. It also contains debugger and breakpoints features. And for the graphical representation like flowchart draw.io is used. It is application that helps to create the diagram with in web browser. It is use to create and edit the diagram of flowchart, ER diagram etc. and for the final report was done in Microsoft word 2016. it is a component of the Microsoft Office suite of productivity software, but can also be purchased as a stand-alone product.

The main goal of this project is to create a full functioning 8-bit binary adder by using the logic gates and handle all the error and exception to the program and not letting the program to letdown. And this is the part which should be done in IDLE for the coding part. So, flowchart illustrating visual representation of the program of bit adder is made to show the flowchart of the program. Now the pseudocode was written to understand the code and it is the simpler way and know its functionality. And for the testing output result is correct or not while the user inputs the valid or invalid inputs and make the testing was successful or not.

While doing this coursework it was difficult in coding and for making 8-bit adder. But after learning and by research I came to know it and implement it properly I faced a problem while doing in flowchart, pseudocode and model of the program. I learned by doing a research and was able to complete coursework.

Model:

1-bit adder circuit:

In this program, we were advised to create an 8-bit adder, by using different logical gates this program calculate the bit adder. The whole bit adder program is showed below given circuit of different logical gates. Figure shown bellow of 8-bit adder circuit implemented for this program:

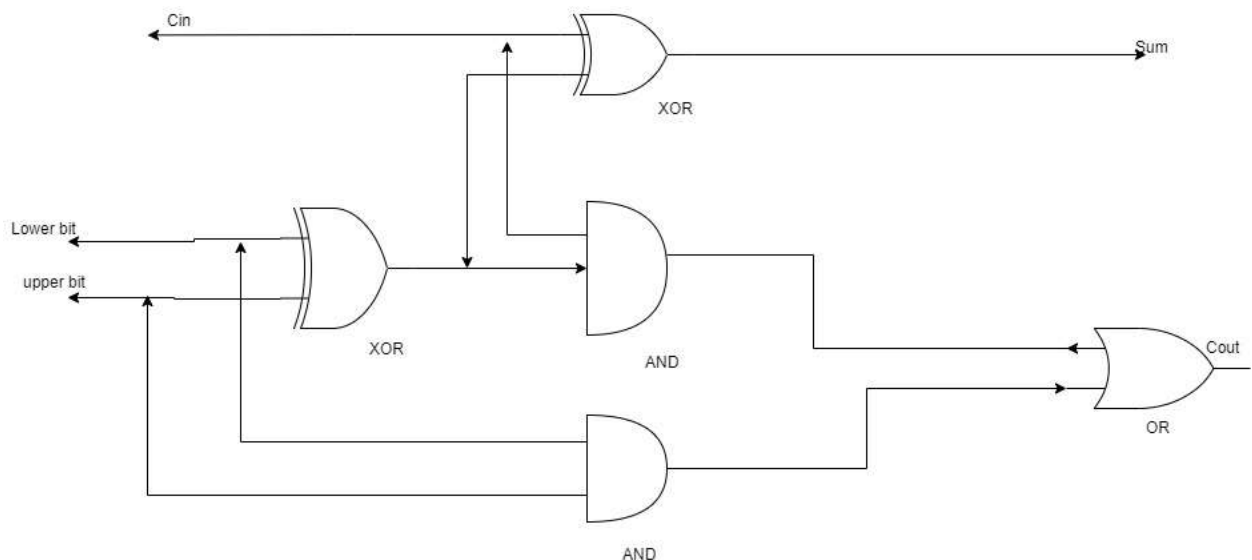


Figure 1 Full adder circuit

So, shown in the above shown diagram, first of all it takes the two input of binary digits and it passes through XOR gates as mentioned above. And the output received from XOR gates and it takes carry from the previous sum and again it is passed through XOR gate. So, by this we have the sum of 1-bit. Since we are adding the binary number there will always be carry over so first it should be calculated to be in the next 1-bit adder, in order to add the carry-over, it should pass the binary digits to AND gates and XOR gate. And after the output obtained by passing XOR gate and carry over from previous 1-bit is again passed through AND gate and finally the final result passes from the passing the initial digits from first

AND gate and second gate by passing through the result of XOR gate and carry over of the previous bit and the sum is again passed through OR gate and the final result carry over from the next 1-bit sum. So by this total process it completes of 1-bit adder but we are asked to do of 8-bit adder so it should be in loop for the 8th times. So, by this we can calculate of 8-bit binary digits.

So, the performed gates of 0-bit adder where AND gate, OR gate, XOR gate. So, by the concept of logical gate, AND gate is the logical gate which returns 1 if the both first and second number is 1 else it returns 0.

OR gate is the logical gate which returns 0 if the both first and second number is 0 else it returns 1. XOR gate is the logical gate which returns 0 if the both first and second number is 1 or 0 else it returns 1.

Lower Bit	Upper Bit	Carry in	sum	Carry out
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

Table 1 Truth table

So, above drawn truth table of a bit adder showing has shown ever possible addition in 8-bit adder and its outputs. So, when both lower bit and upper bit is 0 and it added then the sum is also 0 and its carry out is also 0. And from the previous addition there is now 0,0 and 1 so the sum is 1 and there is no carry out. While in the third when 0 and 1 is added the output, sum is 1 and there is no any carry out. In the fourth, we already have 1 carry in so now we should add 0,1 and the carry in from the previous so we get the sum 10 but 0 remains in the sum and the 1 goes in carry out. So, we have no any carry in so addition of the given number 1 and 0 which sum is 1 with no any carry over. So again, there is 1 carry in and the input is 1 and 0 so the sum is 0 with carry out 1. Again 1 and 1 is added whose outputs comes in 0 with carry out as 1. So, in last 1 and 1 is added with 1 carry in and the sum is 1 and its carry out is 1.

8-bit parallel circuit

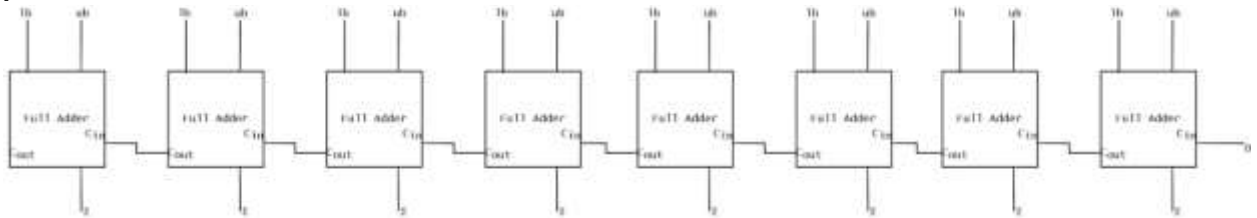


Figure 2 8-bit parallel circuit

So, the figure is of 8-bit adder circuit which is in parallel form. The above shown figure represents the process of addition in 8-bit adder. So, in first it calculates of 1-bit and it run in the loops. So, the process to run in the loop is that the carry over from the previous bit and it is sent to next full adder as a carry out and the digit and the carry over is added and again the carry-in is added to another full adder process. It continues in the loop for the 8 times and the above shown diagram shows all the 8-bit binary addition process.

Algorithm:

An algorithm is the set of steps by steps procedure which define the solution. It involves with several continuous steps. It is generally created to understand the code easily and also it can be done in other programming language.so here is the algorithm of the program.

numValidation model:

- STEP 1: START
- STEP 2: Define function ValidationNum with the parameter num
- STEP 3: If input is less than 255 and greater than 0. It converts in binary form and addition of it.
- STEP 4: If input is less than 0 and greater than 255. This program supports 8-bit addition. please! enter valid number (0-255)
- STEP 5: return num
- STEP 6: STOP

binaryConversion model

- STEP 1: START

STEP 2: Define function binary conversion with the parameter decimal number

STEP 3: convert input number into binary using conversion module

STEP 4: return bit

STEP 5: STOP

Addition model:

STEP 1: START

STEP 2: define function orGate with the parameter(x,y)

STEP 3: return x and y

STEP 4: define function andGate with the parameter (x,y)

STEP 5: return X and y

STEP 6: define function XORgate with parameter (x,y)

STEP 7: return x and y

STEP 8: define binary addition with the parameter of (number1, number2)

STEP 9: assign number into string

STEP 10: for l in range (0, (8-len(str1)), +1):

STEP 11: for l in range (0, (8-len(str2)), +1):

STEP 12: assign value of sum and carry

STEP 13: if carry is equal to 0

STEP 14: invoking sum

STEP 15: DISPLAY Sorry! This program only supports 8-bit addition

STEP 16: STOP

Reverse model

STEP 1: START

STEP 2: define function reverse with the parameter (number list)

STEP 3: Subtract 1 from each parameter

STEP 4: Binary.append(NumberList[i])
STEP 5: BinaryNum=BinaryNum+str(NumberList[i])
STEP 6: return BinaryNumber
STEP 7: STOP

Main model

STEP 1: START
STEP 2: Display “!!Welcome to program!!”
STEP 3: assign loop as true
STEP 4: run the code till the loop is true
STEP 5: if it arises false than
STEP 6: input first decimal number
STEP 7: input second decimal number
STEP 8: if input is out of number validation then display invalid input
STEP 9: if it is true then DISPLAY sum in binary
STEP 10: DISPLAY Do you want to continue this application?? (Type no to exit) "
STEP 11: if the program gets no as a input it breaks the loop
STEP 12: STOP

Pseudocode

Main program module:

Define headline

Display “!!Welcome to the program”

Invoking Headline

Assign loop as true

Assign while loop as true

Assign first number as false

Assign while first number as false

Try:

Display Enter first decimal number

Assign first number as false

Expect:

Display "invalid input"

Assign second number as false

Assign while second number as false

Try:

Display Enter second decimal number

Assign second number as true

Expect

Display invalid input

First number is equal to Binary conversion of first number

Second number is equal to Binary conversion of second number

Binary first number is equal to reverse first number

Binary second number is equal to reverse second number

Binary addition is equal to the addition of first binary number and second binary number

Display binary number of first number

Display binary number of second number

Display addition of first binary number and second binary number

Display "Do you want to continue this application?? (Type no to exit) "

If it assigns as no

Break

Define Binary conversion

Bit= []

Counter=0

Assign counter as 8 times

While remainder is equal to the division of decimal number by 2

Appending the remainder in bit

Dividing the conversion of decimal by 2

Counter is equal to the addition of it's in each loop

Invoking bit

Define Validation Num

Assign number is greater than 0 and less than 255

If it is equal to false

Display Input invalid

Display ("please! enter valid number (0-255): ")

Invoking num

Define xor gate (x,y)

Invoking x and y

Define and gate

Invoking x and y

Define OR gate

Invoking x and y

Define binary addition with parameter of number1 and number2

Assign str as number1

Assign str2 as number 2

Converting it to 8-digit byte

For l in range (0,8-return str1 and adding 1 str1 and "0")

Str1 is equal too addition carry and str1

For l in range (0,8-return str2 and adding 1 str1 and "0")

Str2 is equal to addition of carry and str2

Assign the value of sum and carry

For l in range (7, -1, -1)

The value af x and y is equal to str1 and str2

The final sum is the addition of xor carry and the sum

If carry is equal to 0

Invoking sum

Else

Display Sorry! The number is greater in the byte...

Define reverse with parameter (number list)

Binary = []

Binary number = ""

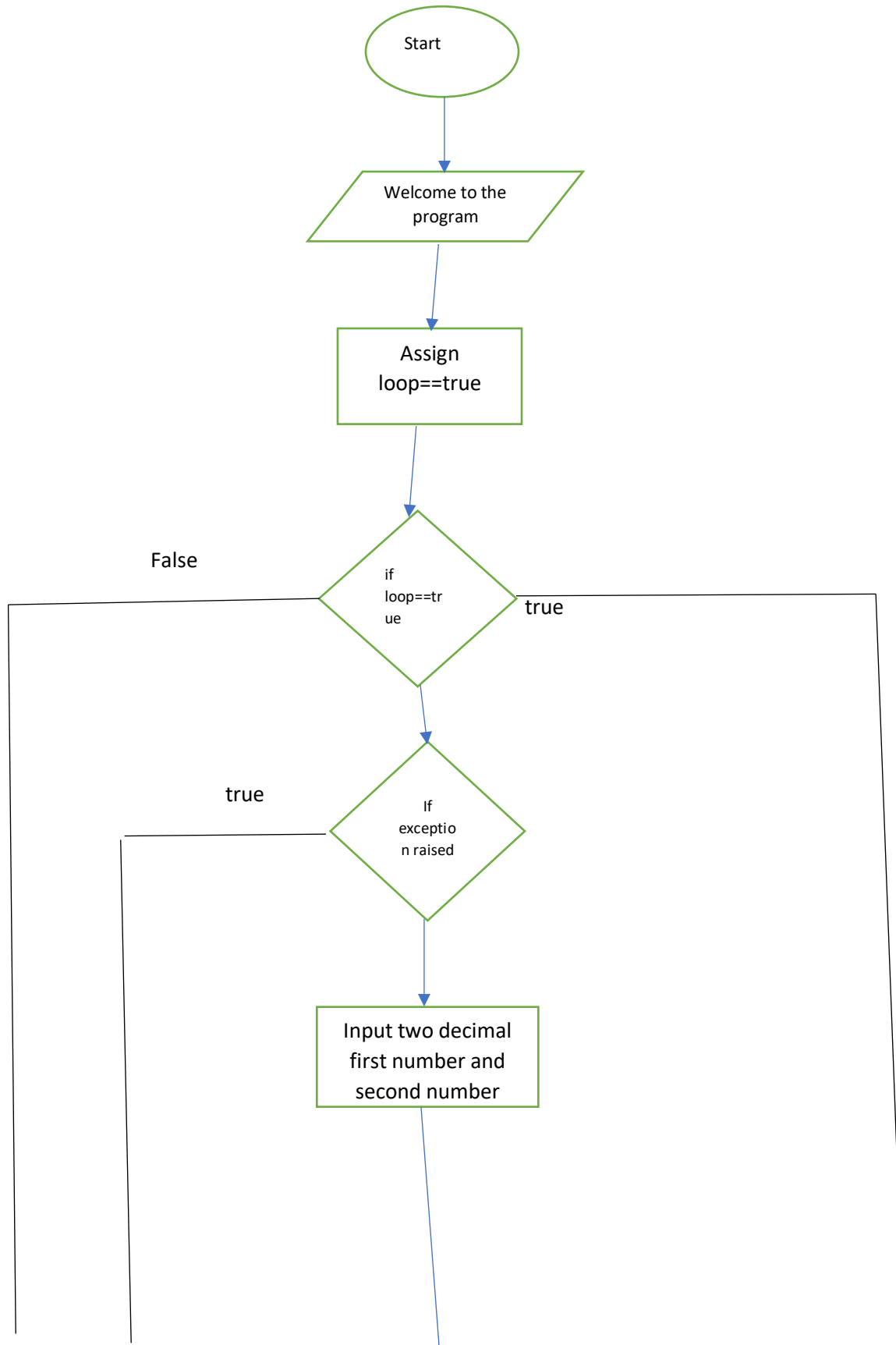
For l in range of length of number list

Appending the binary of number list

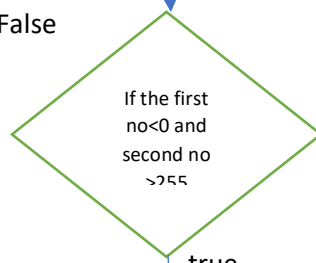
Binary number is equal to the addition of binary list and number list

Invoking binary number

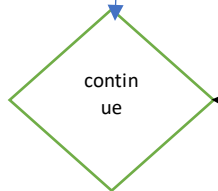
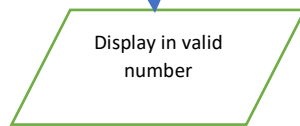
Flowchart:



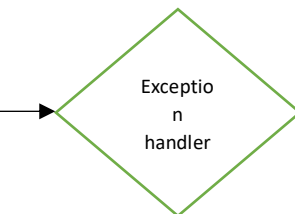
False



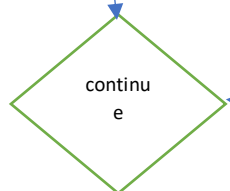
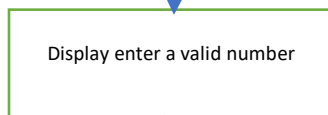
true



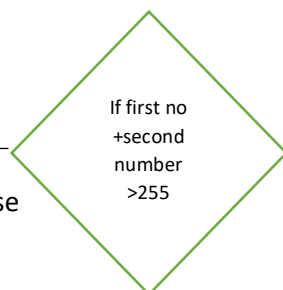
Exception handler



true



false



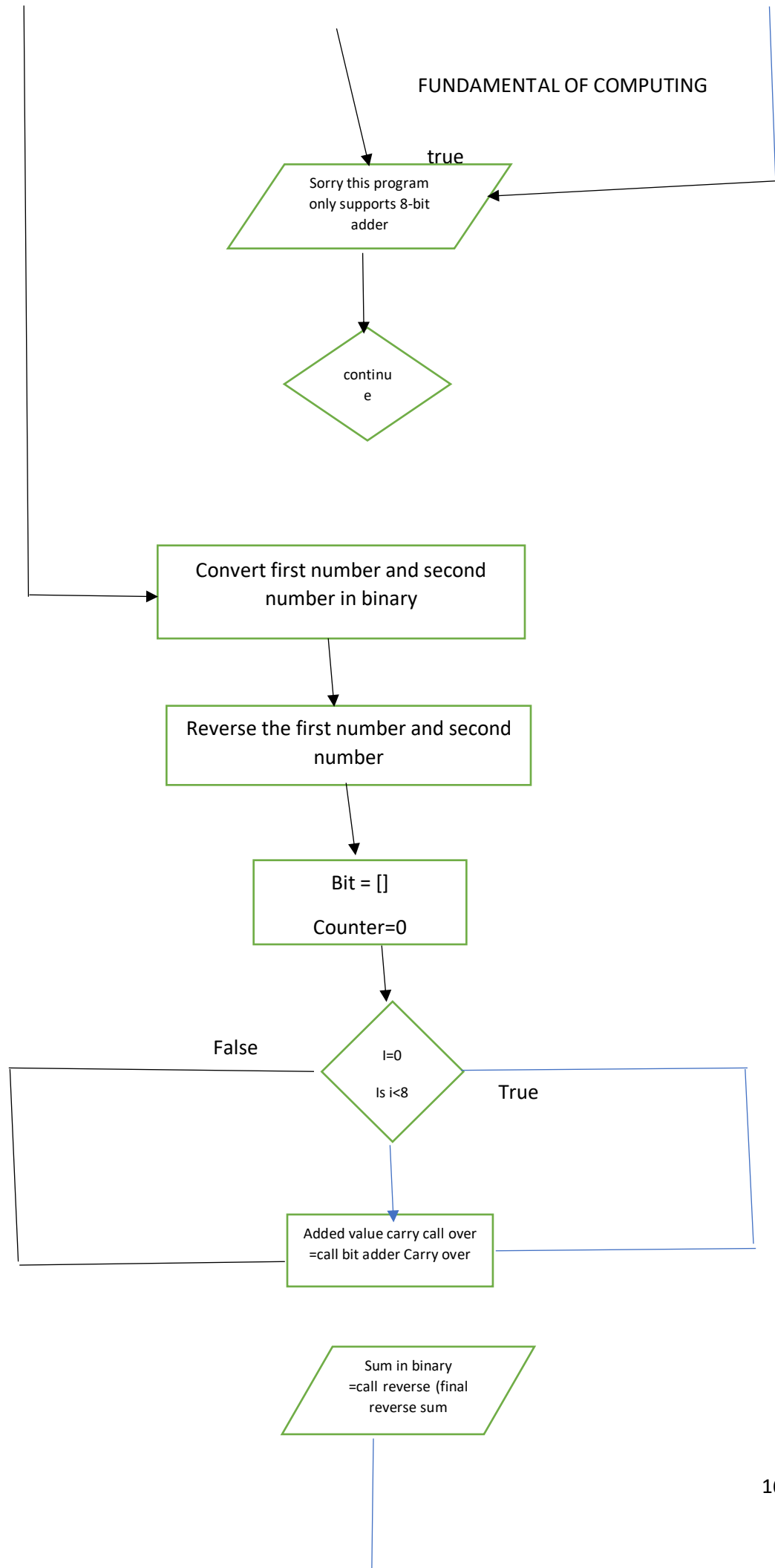
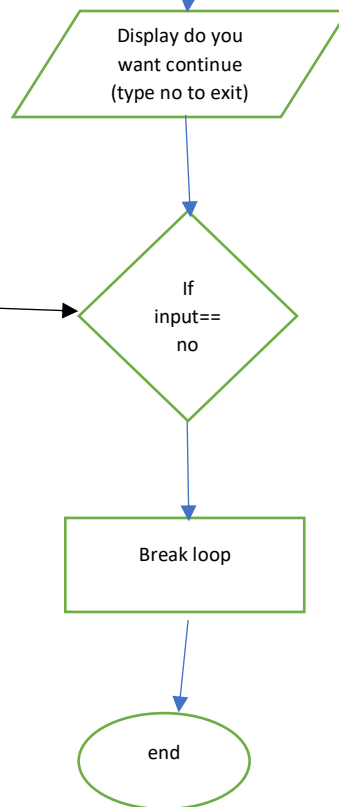


Figure 3 Flowchart



Data Structure:

Data structure is basically a structure which holds some data together or in other word it is used to store collection of data. It stores and process data in extra ordinary speed. So, the stored data is highly processed securely and alco can be accessed easily. The processing of data can be processed in the shortest time possible but without compromising the accuracy. In the processing of the data the data structure deals in the data structed and it processes. The important thing is that the data stored in the disk as permanent storage is not referred as a data structure.

So, in this program I have used a data structure which are list, integer, String etc. I have used the list data structure because for the implementation of 8-bit adder I need to give input in binary form and add ever digit bit by bit because the give input can be accessible by index. So, in list we can give the input value in both positive, negative and also in String. So, for the index operator "[]" is used. Integer is the next data structure

that is used while making of the program. An integer represents the mathematical number as the program is all about 8-bit adder so we need the calculate the mathematical addition. String is the next data structure used in the program. it is used for the sequence of the character surrounded by either single quotation marks, or double quotation marks. It Is used to display message so that the user of the program can be easy the understand.

There are also others data structure which was not stored while making this program like as tuples, set, dictionaries etc. I have not used tuples because in this program we don't have to add elements in the list. Like as dictionary is also not used in this program because in this program order is necessary but the data structure Dictionary is used order of data is necessary. So, the set of unordered collection always contain unique item. So, the above-mentioned data structure is not required in this program.

Testing

Test no 1

Table 2: Test 1 details

Test no:	1
Objective:	To run the program successfully
Action:	First decimal number: 5 Second decimal number: 5
Expected Result:	To give the binary addition of 5 and 5
Actual Result:	Binary addition done successfully
Conclusion:	The test is successful.

Table 2 test 1

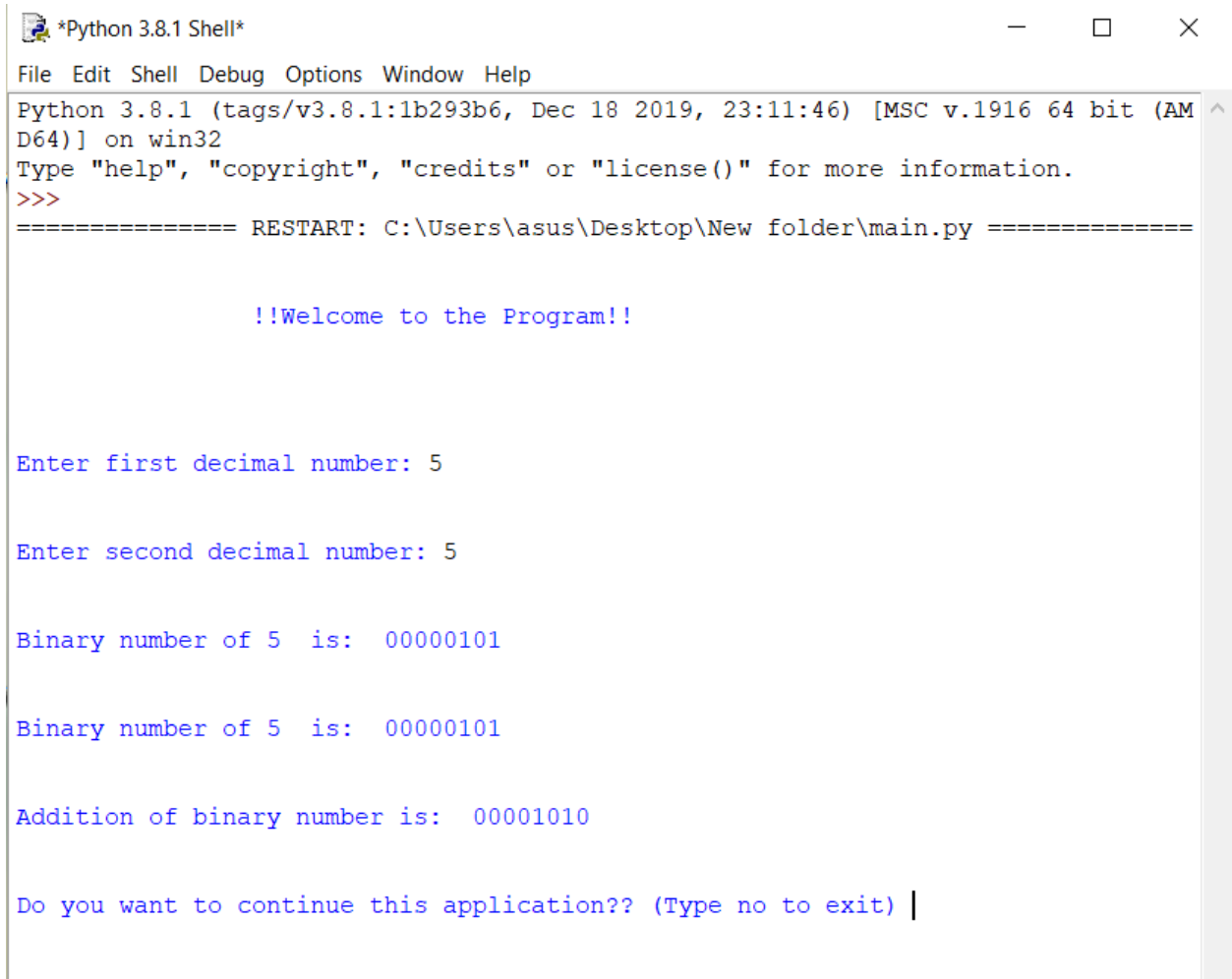
 *Python 3.8.1 Shell*
File Edit Shell Debug Options Window Help
Python 3.8.1 (tags/v3.8.1:1b293b6, Dec 18 2019,
D64)] on win32
Type "help", "copyright", "credits" or "license
>>>
===== RESTART: C:\Users\asus\Desktop\

!!Welcome to the Program!!

Enter first decimal number: 5

Enter second decimal number: 5|

Figure 4 input value



```

Python 3.8.1 Shell
File Edit Shell Debug Options Window Help
Python 3.8.1 (tags/v3.8.1:1b293b6, Dec 18 2019, 23:11:46) [MSC v.1916 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
===== RESTART: C:\Users\asus\Desktop\New folder\main.py =====

    !!Welcome to the Program!!

Enter first decimal number: 5

Enter second decimal number: 5

Binary number of 5 is: 00000101

Binary number of 5 is: 00000101

Addition of binary number is: 00001010

Do you want to continue this application?? (Type no to exit) |

```

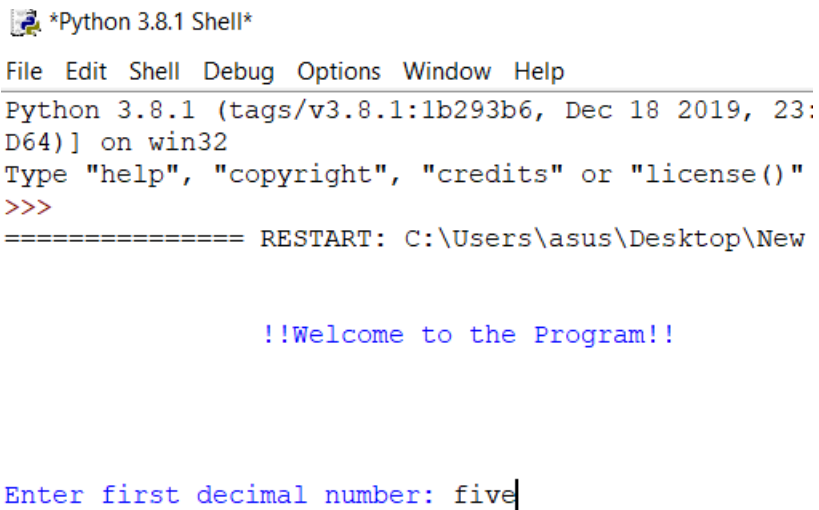
Figure 5 result of test 1

Test no 2:

Test no:	2
Objective:	To handle error while a user tries to enter a String instead of integer To show the error while entering string data type instead of integer

Action:	First decimal number: "five"
Expected Result:	To display popup message
Actual Result:	Display popup message
Conclusion:	The test is successful.

Figure 6 test no 2



```

*Python 3.8.1 Shell*
File Edit Shell Debug Options Window Help
Python 3.8.1 (tags/v3.8.1:1b293b6, Dec 18 2019, 23:
D64)] on win32
Type "help", "copyright", "credits" or "license()"
>>>
===== RESTART: C:\Users\asus\Desktop\New

!!Welcome to the Program!!

Enter first decimal number: five

```

Figure 7 input value

```

Python 3.8.1 Shell*
File Edit Shell Debug Options Window Help
Python 3.8.1 (tags/v3.8.1:1b293b6, Dec 18 2019, 23:11:46)
D64) on win32
Type "help", "copyright", "credits" or "license()" for mor
>>>
===== RESTART: C:\Users\asus\Desktop\New folder\

    !!Welcome to the Program!!

Enter first decimal number: five

Invalid Input

Enter first decimal number: |

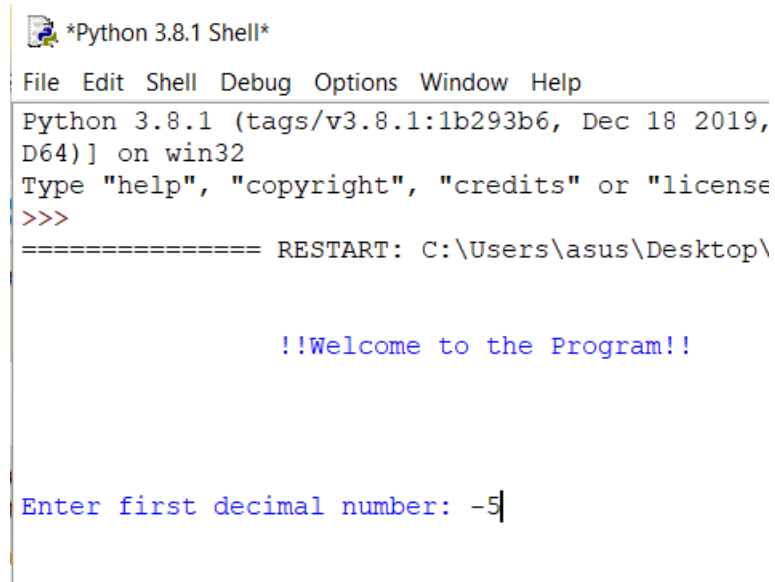
```

Figure 8 result of test no 2

Test no 3

Test no:	3
Objective:	To handle error while a user tries to enter a number other than 0-255
Action:	First decimal number: -5 Second decimal number: 5
Expected Result:	To display message
Actual Result:	Display message
Conclusion:	The test is successful.

Table 3 test no 3



```

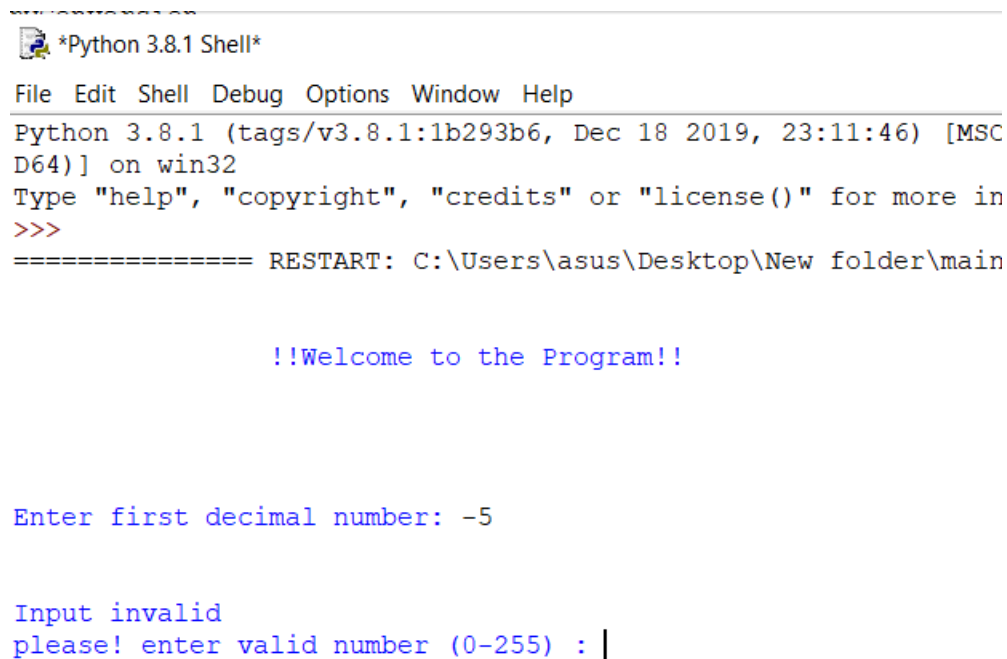
Python 3.8.1 Shell
File Edit Shell Debug Options Window Help
Python 3.8.1 (tags/v3.8.1:1b293b6, Dec 18 2019,
D64)] on win32
Type "help", "copyright", "credits" or "license
>>>
===== RESTART: C:\Users\asus\Desktop\

!!Welcome to the Program!!

Enter first decimal number: -5

```

Figure 9 input value



```

Python 3.8.1 Shell
File Edit Shell Debug Options Window Help
Python 3.8.1 (tags/v3.8.1:1b293b6, Dec 18 2019, 23:11:46) [MSC
D64)] on win32
Type "help", "copyright", "credits" or "license()" for more in
>>>
===== RESTART: C:\Users\asus\Desktop\New folder\main

!!Welcome to the Program!!

Enter first decimal number: -5

Input invalid
please! enter valid number (0-255) : |

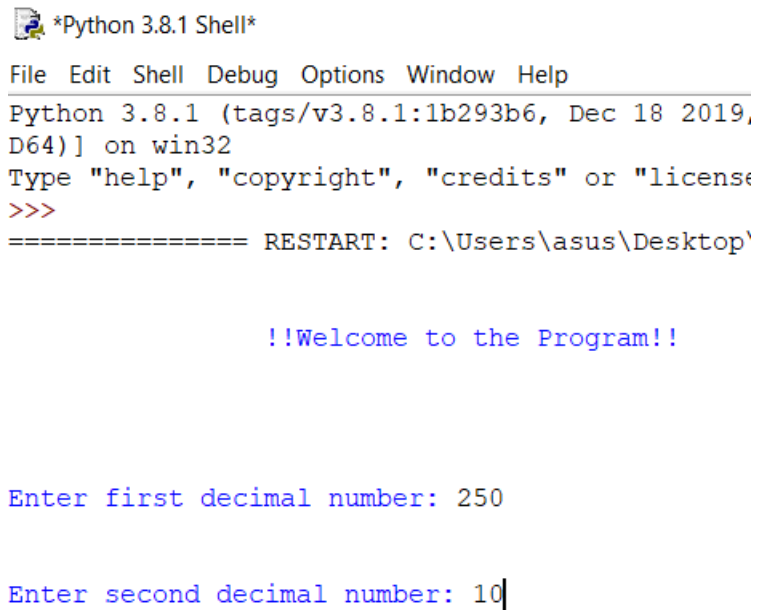
```

Figure 10 result of test no 3

Test no 4

Test no:	4
Objective:	To handle error while a user tries to enter a number combined more than 255
Action:	First decimal number: 250 Second decimal number: 10
Expected Result:	To display a message
Actual Result:	Display message shown

Table 4 test no 4



```

Python 3.8.1 Shell*
File Edit Shell Debug Options Window Help
Python 3.8.1 (tags/v3.8.1:1b293b6, Dec 18 2019, D64) on win32
Type "help", "copyright", "credits" or "license()"
>>>
===== RESTART: C:\Users\asus\Desktop\

!!Welcome to the Program!!

Enter first decimal number: 250

Enter second decimal number: 10|
  
```

Figure 11 input value of test 4


```

Python 3.8.1 Shell*
File Edit Shell Debug Options Window Help
Python 3.8.1 (tags/v3.8.1:1b293b6, Dec 18 2019, 23:11:46) [MSC v
64)] on win32
Type "help", "copyright", "credits" or "license()" for more info
>>>
===== RESTART: C:\Users\asus\Desktop\New folder\main.py

!!Welcome to the Program!!

Enter first decimal number: 250

Enter second decimal number: 10

Sorry!this program only supports 8-bit addition

Binary number of 250 is: 11111010

Binary number of 10 is: 00001010

Addition of binary number is: None

Do you want to continue this application?? (Type no to exit) |

```

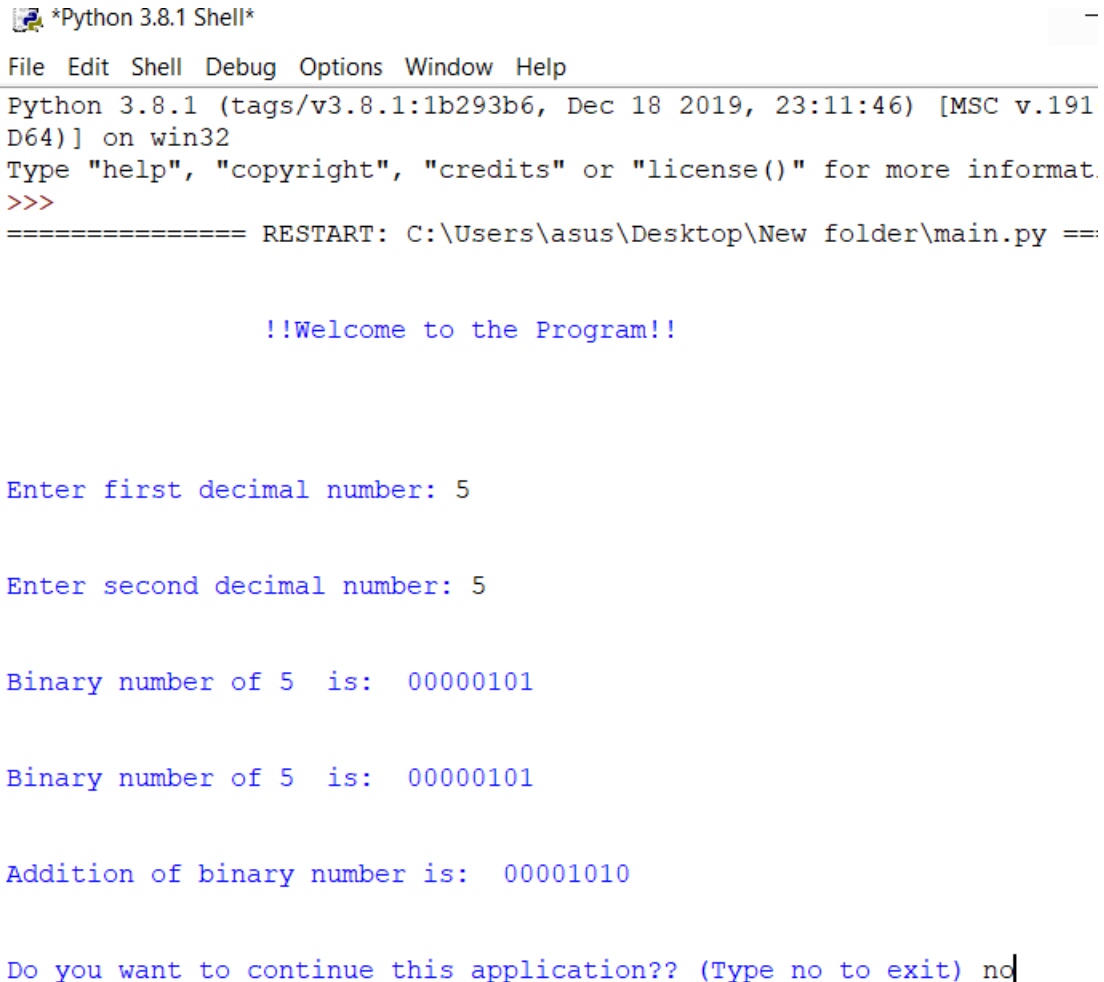
Figure 12 result of test 4

Test no 5

Test no:	5
Objective:	To terminate the program as per user wants
Action:	Enter "no" when prompt saying "Do you wish to continue? if not then type 'no' to exit:" appears
Expected Result:	To terminate the program if user says so

Actual Result:	Program terminated
Conclusion:	The test is successful.

Table 5 test no 5



```

Python 3.8.1 Shell
File Edit Shell Debug Options Window Help
Python 3.8.1 (tags/v3.8.1:1b293b6, Dec 18 2019, 23:11:46) [MSC v.191
D64)] on win32
Type "help", "copyright", "credits" or "license()" for more informat
>>>
===== RESTART: C:\Users\asus\Desktop\New folder\main.py =====

!!Welcome to the Program!!

Enter first decimal number: 5

Enter second decimal number: 5

Binary number of 5 is: 00000101

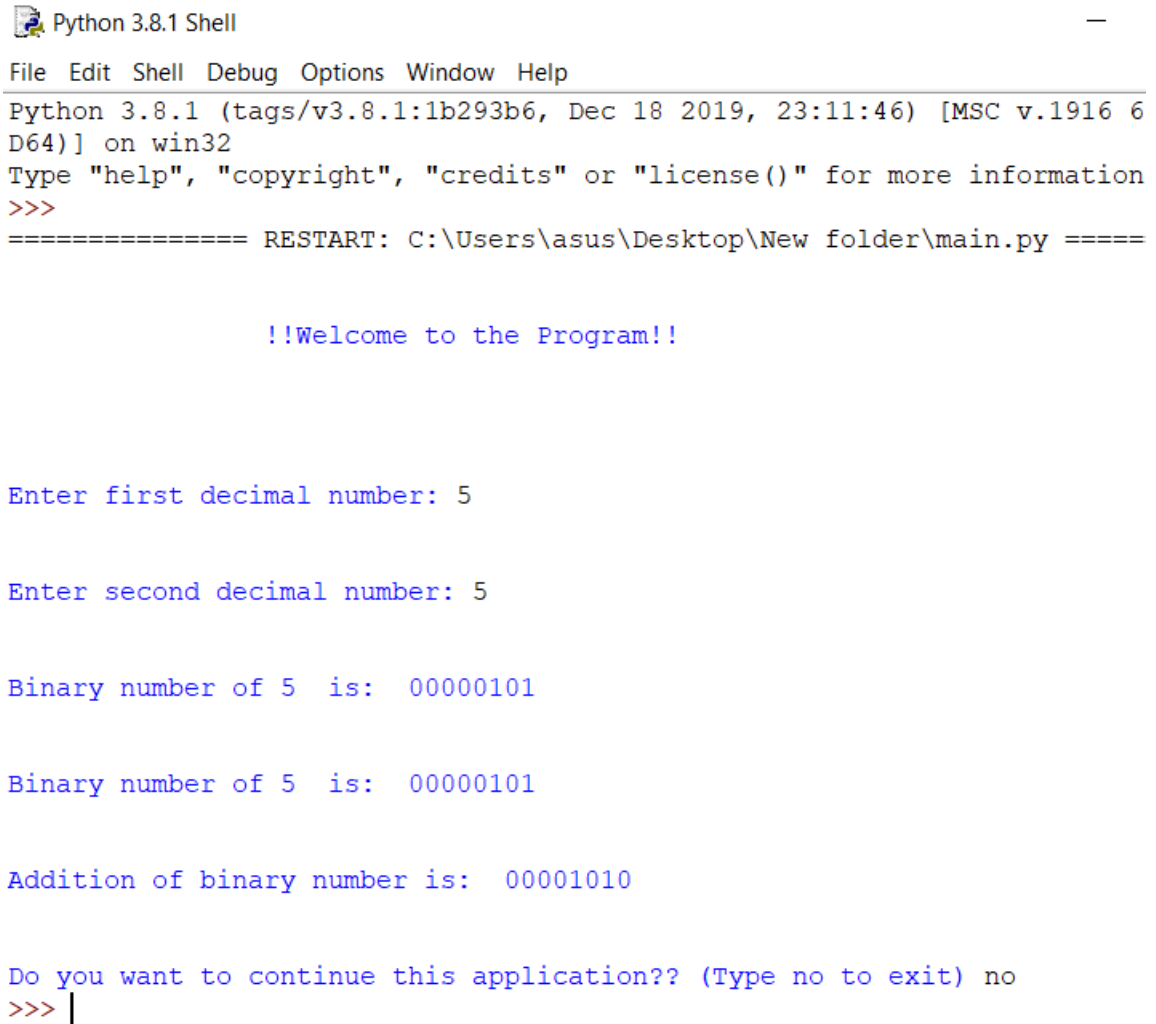
Binary number of 5 is: 00000101

Addition of binary number is: 00001010

Do you want to continue this application?? (Type no to exit) no

```

Figure 13 input value of test 5



A screenshot of a Python 3.8.1 Shell window. The window title is "Python 3.8.1 Shell". The menu bar includes "File", "Edit", "Shell", "Debug", "Options", "Window", and "Help". The status bar shows "Python 3.8.1 (tags/v3.8.1:1b293b6, Dec 18 2019, 23:11:46) [MSC v.1916 6 D64)] on win32". The main text area displays the following content: a prompt to type "help", "copyright", "credits" or "license()" for more information, followed by a restart command for a file named "main.py" located at "C:\Users\asus\Desktop\New folder\main.py". The program then prints "!!Welcome to the Program!!". It prompts for a "first decimal number" (5) and a "second decimal number" (5). It then displays the binary representation of 5 as "0000101" twice. Next, it shows the "Addition of binary number is: 00001010". Finally, it asks "Do you want to continue this application?? (Type no to exit)" and the user has entered "no". The prompt ">>>" is visible at the end of the last line.

```
Python 3.8.1 Shell
File Edit Shell Debug Options Window Help
Python 3.8.1 (tags/v3.8.1:1b293b6, Dec 18 2019, 23:11:46) [MSC v.1916 6
D64)] on win32
Type "help", "copyright", "credits" or "license()" for more information
>>>
===== RESTART: C:\Users\asus\Desktop\New folder\main.py =====

!!Welcome to the Program!!

Enter first decimal number: 5

Enter second decimal number: 5

Binary number of 5 is: 0000101

Binary number of 5 is: 0000101

Addition of binary number is: 00001010

Do you want to continue this application?? (Type no to exit) no
>>> |
```

Figure 14 result of test 5

Conclusion

This coursework was first coursework of module Fundamentals of computing were the given project. It is all about creating 8-bit binary adder by using the concept of logical gates. To not get the error in the program it should be manage in coding part. So, it is done in coding part and for the report or documentation we are advised to make pseudocode of the coding, flowchart of the coding, algorithm of coding. The main goal of this program is to create the working of 8-bit binary adder using logical gates.

Firstly, it was difficult while doing in the coursework. It was not possible to build whole task in a single try or a single day. to complete every single question from this coursework was high from my comfort level. At first, I was even not able to understand the question properly when it was published but when I started to read the coursework and discussed with the teacher and my fellow friends then slowly it starts to make sense. Firstly, I researched about the whole coursework and by knowing the question I started to do it and finally completed it.