# Assignment #4

## — Make a Binary Calculator in Python Code —

```python
import sys

input_number = int(float(sys.argv[1]))

if not (input_number >= 0 and input_number < 256):
    print("Your number is not within the range of 0 to 255 inclusive!")
    sys.exit()

current_number = ""

current_number = str(int(input_number % 2)) + current_number
input_number = input_number / 2

current_number = str(int(input_number % 2)) + current_number
input_number = input_number / 2

current_number = str(int(input_number % 2)) + current_number
input_number = input_number / 2

current_number = str(int(input_number % 2)) + current_number
input_number = input_number / 2

current_number = str(int(input_number % 2)) + current_number
input_number = input_number / 2

current_number = str(int(input_number % 2)) + current_number
input_number = input_number / 2

current_number = str(int(input_number % 2)) + current_number
input_number = input_number / 2

current_number = str(int(input_number % 2)) + current_number
input_number = input_number / 2

print(current_number)
```

## — Step 4: Testing Bad Input (Actual Results) —

- Finish your testing table from Assignment #3
    - What are the actual results from testing with bad inputs?
        - When I tried with a negative number it gave me a reverse buffer overflow error. -1 gave 11111111 as a result.
        - When I tried a decimal number it said there was an invalid literal for int() with base 10
    - Do the actual results match match what you expected?
        - The actual results do act how I expected when coding an error check.

## — Design Error Handling —

I included it above. If the input is good, it will do the program. If the input is not good, it will print that the number is not within the range and then exits the program.

Here is the syntax for python:

```
if not (input_number >= 0 and input_number < 256):
    print("Your number is not within the range of 0 to 255 inclusive!")
    sys.exit()
```