# Assignment 3 – Control Flow

## Control Flow

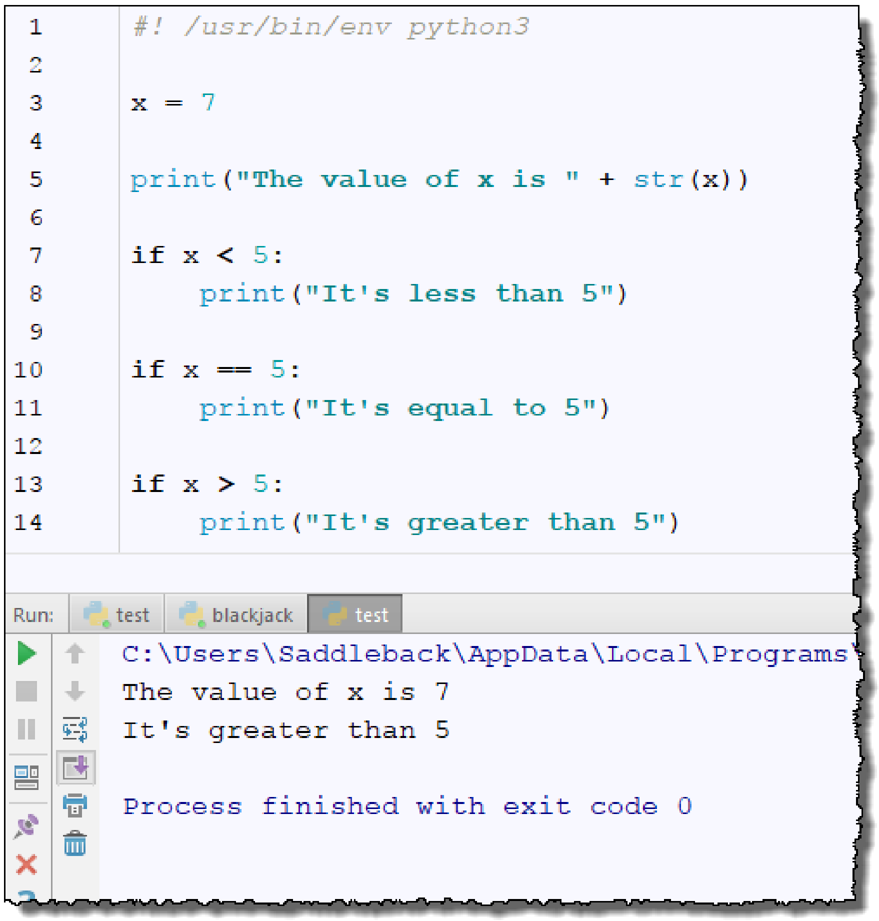
When creating a program in any programming language, you will need to control flow using binary expressions like the *if statement* and looping control flow like the *for statement*.

### Binary Expressions

#### *if Statements*

Binary (or Boolean) expressions are expressions that evaluation to True or False.

1. Code the following
   Screen Capture #1 (1 point)



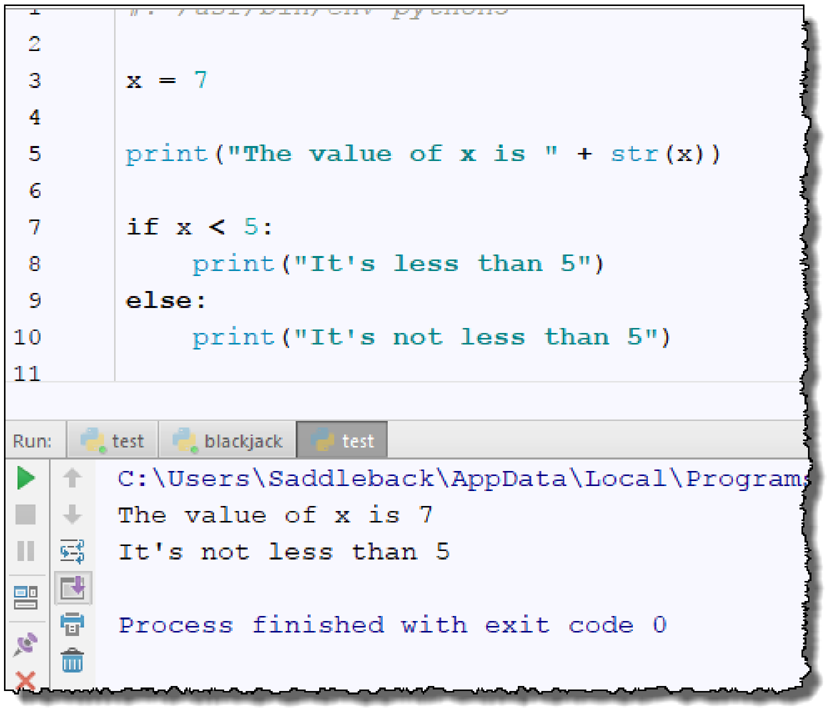
```
#! /usr/bin/env python3

x = 7

print("The value of x is " + str(x))

if x < 5:
    print("It's less than 5")

if x == 5:
    print("It's equal to 5")

if x > 5:
    print("It's greater than 5")
```

```
C:\Users\Saddleback\AppData\Local\Programs
The value of x is 7
It's greater than 5

Process finished with exit code 0
```

*if else Statements*

2. Code the following

   Screen Capture #2 (1 point)

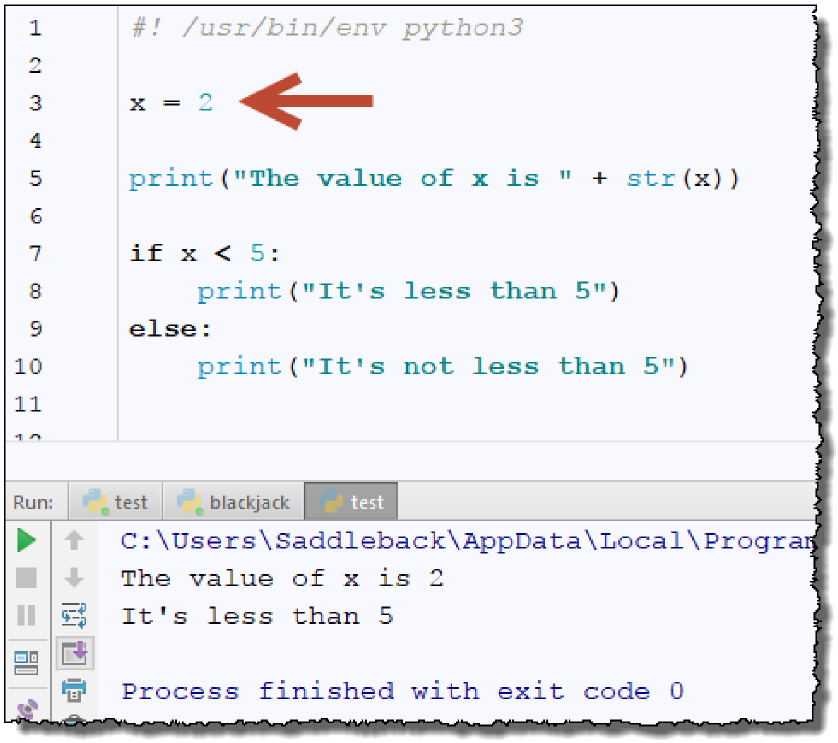


```
#! /usr/bin/env python3

x = 7

print("The value of x is " + str(x))

if x < 5:
    print("It's less than 5")
else:
    print("It's not less than 5")
```

```
C:\Users\Saddleback\AppData\Local\Program
The value of x is 7
It's not less than 5

Process finished with exit code 0
```

3. Change the value of x

   Screen Capture #3 (1 point)



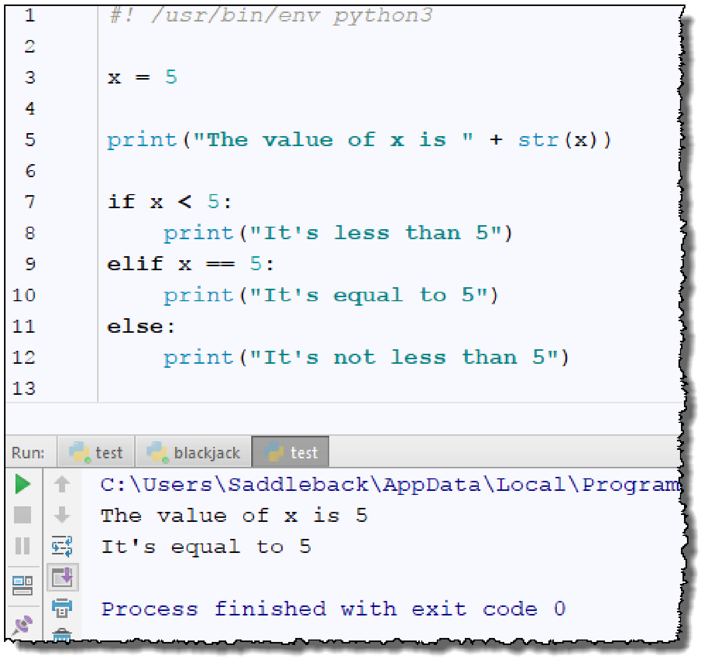
```
#! /usr/bin/env python3

x = 2

print("The value of x is " + str(x))

if x < 5:
    print("It's less than 5")
else:
    print("It's not less than 5")
```

```
C:\Users\Saddleback\AppData\Local\Program
The value of x is 2
It's less than 5

Process finished with exit code 0
```

### *if elif Statements*

4. Code the following

Screen Capture #4 (1 point)



```
#! /usr/bin/env python3

x = 5

print("The value of x is " + str(x))

if x < 5:
    print("It's less than 5")
elif x == 5:
    print("It's equal to 5")
else:
    print("It's not less than 5")

```

```
C:\Users\Saddleback\AppData\Local\Program
The value of x is 5
It's equal to 5

Process finished with exit code 0
```

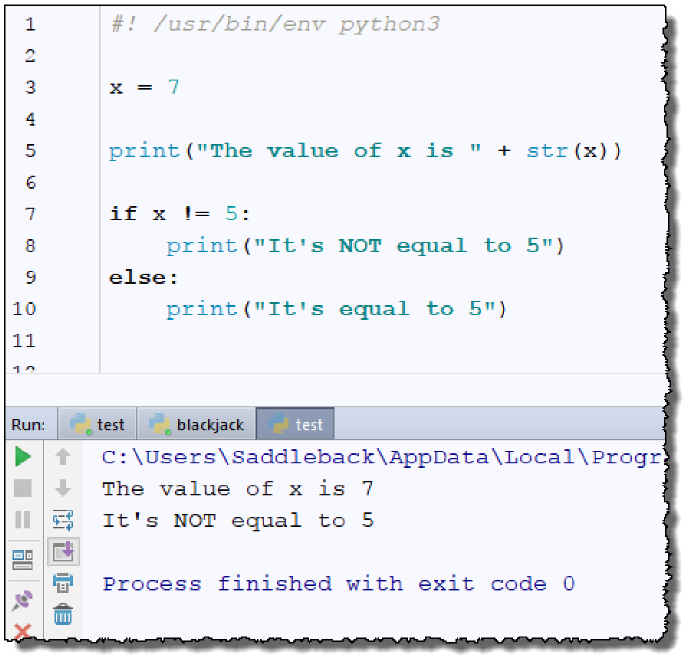
## Logical Operators

Logical operators allow you to combine two or more Boolean expressions

### *NOT (!)*

5. Code the following

Screen Capture #5 (1 point)



```
#! /usr/bin/env python3

x = 7

print("The value of x is " + str(x))

if x != 5:
    print("It's NOT equal to 5")
else:
    print("It's equal to 5")

```

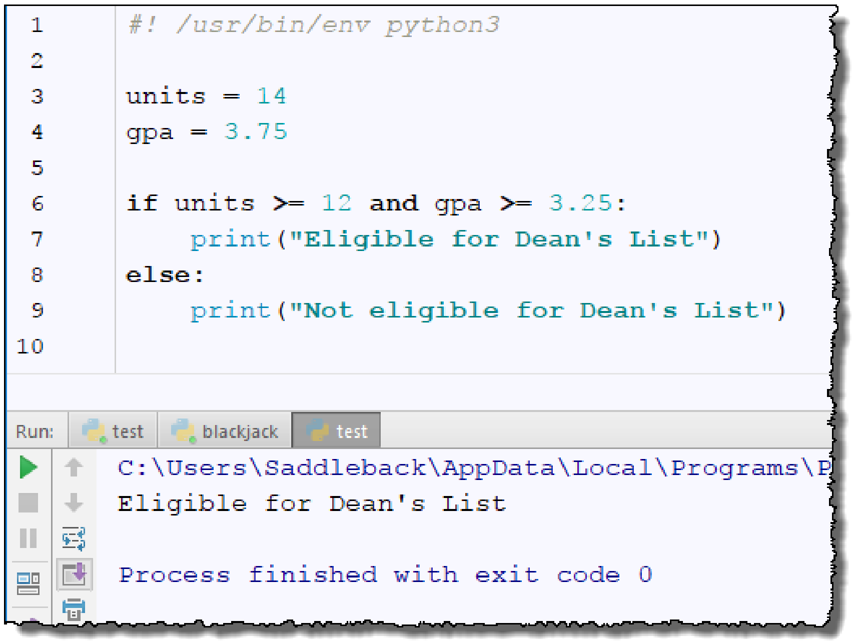
```
C:\Users\Saddleback\AppData\Local\Progr
The value of x is 7
It's NOT equal to 5

Process finished with exit code 0
```

## *AND*

With the *and* operator, the condition on both sides of the operator must be true for the condition to be true

6. Code the following

Screen Capture #6 (1 point)



```
#! /usr/bin/env python3

units = 14
gpa = 3.75

if units >= 12 and gpa >= 3.25:
    print("Eligible for Dean's List")
else:
    print("Not eligible for Dean's List")

```

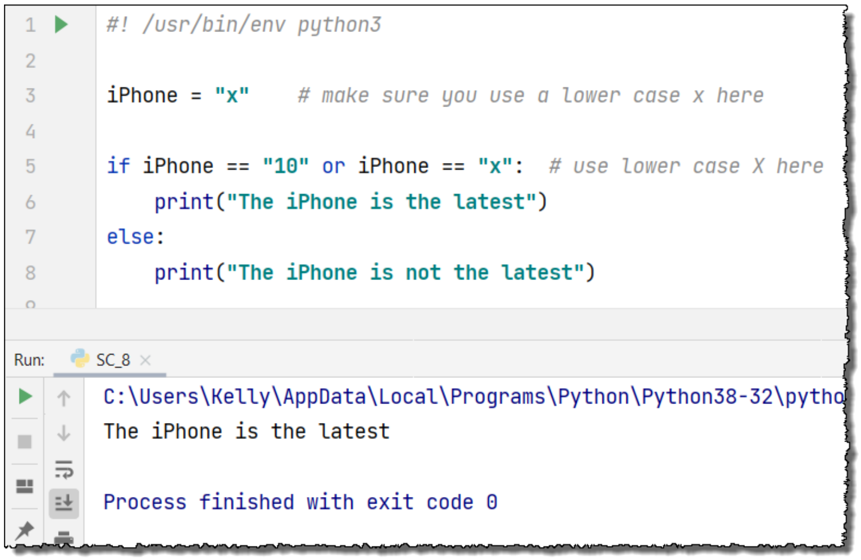
```
C:\Users\Saddleback\AppData\Local\Programs\P
Eligible for Dean's List

Process finished with exit code 0
```

## *OR*

With the *or* operator, only one of the conditions needs be true for the condition to be true

7. Code the following

Screen Capture #7 (1 point)



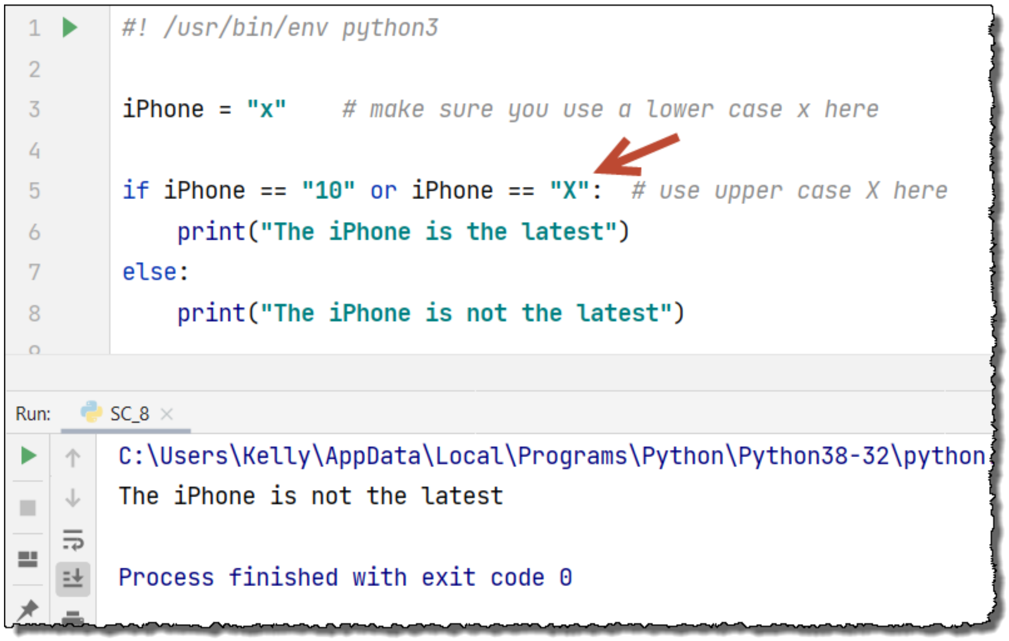
```
#! /usr/bin/env python3

iPhone = "x"    # make sure you use a lower case x here

if iPhone == "10" or iPhone == "x":  # use lower case X here
    print("The iPhone is the latest")
else:
    print("The iPhone is not the latest")
```

```
C:\Users\Kelly\AppData\Local\Programs\Python\Python38-32\pytho
The iPhone is the latest

Process finished with exit code 0
```

## *Comparing Strings*

When comparing string, case matters. To account for this, convert using Python's built-in .upper() or .lower() function.

8. Code the following

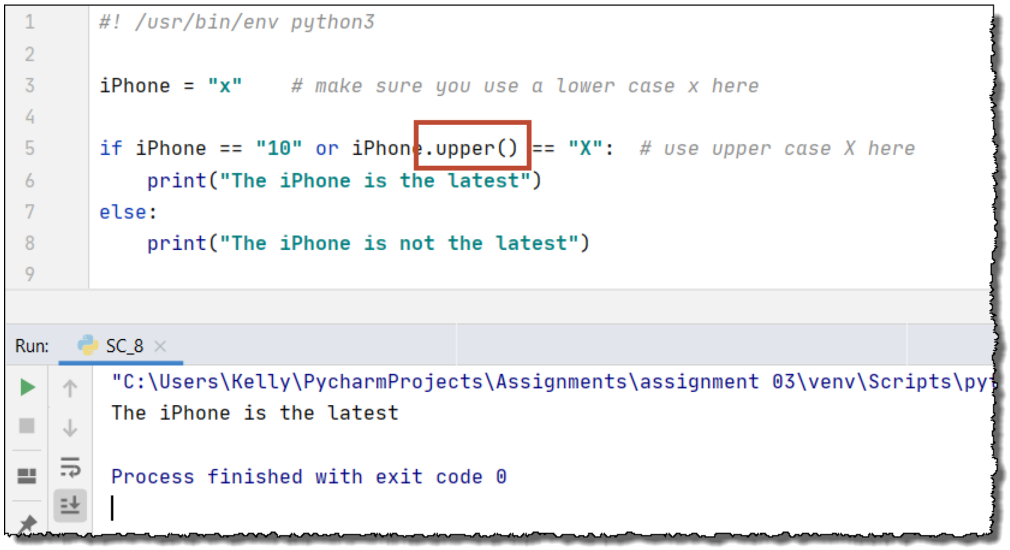


```
#! /usr/bin/env python3

iPhone = "x"     # make sure you use a lower case x here

if iPhone == "10" or iPhone == "X":  # use upper case X here
    print("The iPhone is the latest")
else:
    print("The iPhone is not the latest")
```

Run: SC_8

```
C:\Users\Kelly\AppData\Local\Programs\Python\Python38-32\python
The iPhone is not the latest

Process finished with exit code 0
```

9. Modify the following:
Screen Capture #8 (1 point)



```
#! /usr/bin/env python3

iPhone = "x"     # make sure you use a lower case x here

if iPhone == "10" or iPhone.upper() == "X":  # use upper case X here
    print("The iPhone is the latest")
else:
    print("The iPhone is not the latest")
```

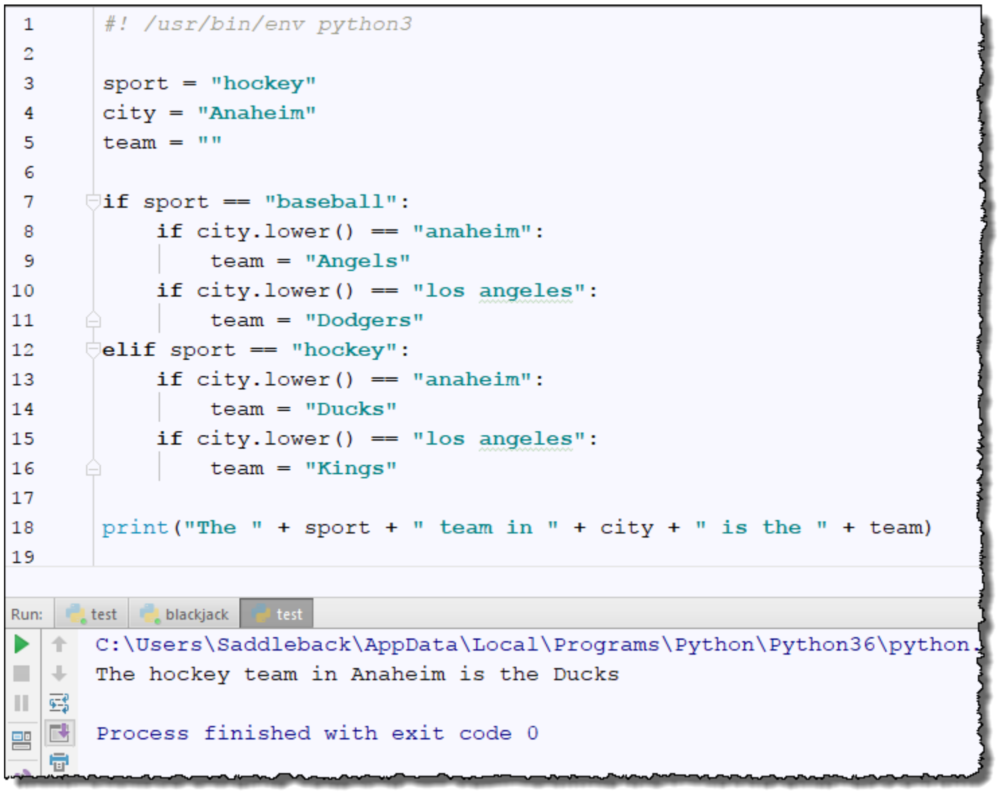
Run: SC_8

```
"C:\Users\Kelly\PycharmProjects\Assignments\assignment 03\venv\Scripts\py
The iPhone is the latest

Process finished with exit code 0
```

## *Nested if Statements*

10. Code the following

Screen Capture #9 (1 point)



```python
#! /usr/bin/env python3

sport = "hockey"
city = "Anaheim"
team = ""

if sport == "baseball":
    if city.lower() == "anaheim":
        team = "Angels"
    if city.lower() == "los angeles":
        team = "Dodgers"
elif sport == "hockey":
    if city.lower() == "anaheim":
        team = "Ducks"
    if city.lower() == "los angeles":
        team = "Kings"

print("The " + sport + " team in " + city + " is the " + team)
```

```
C:\Users\Saddleback\AppData\Local\Programs\Python\Python36\python.
The hockey team in Anaheim is the Ducks

Process finished with exit code 0
```

## Iterative Structure

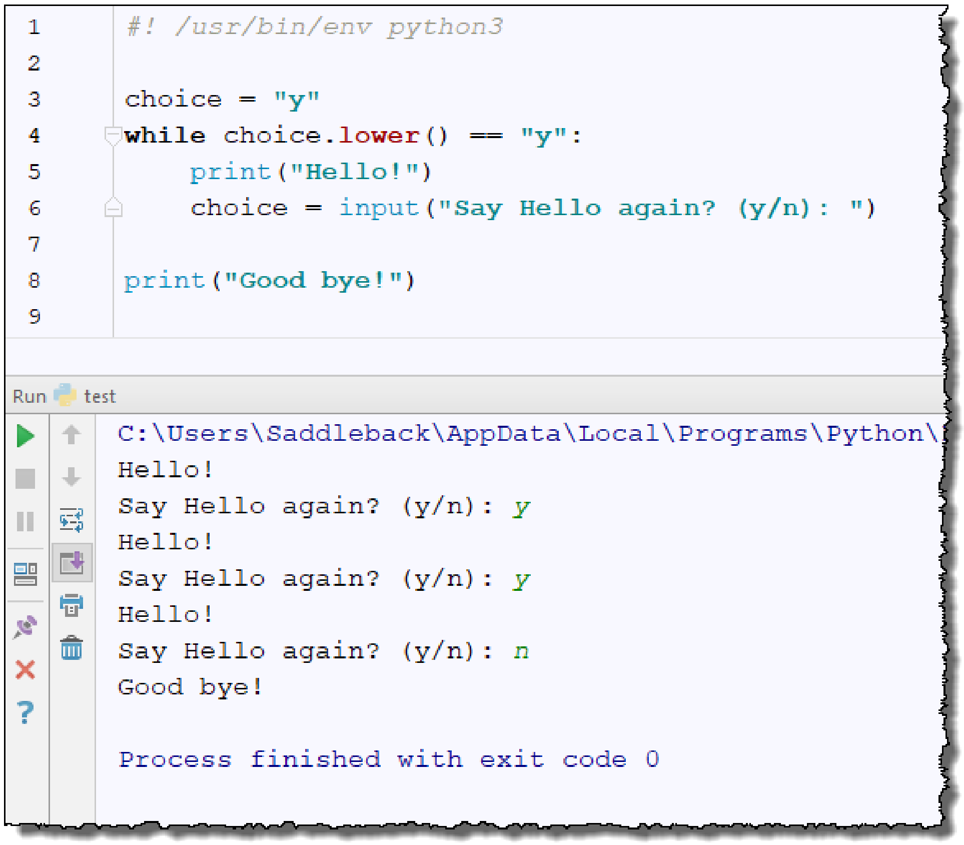
Python iterative code is code that repeats itself using a for and/or while statement

### *while Loop*

Binary (or Boolean) expressions are expressions that evaluation to True or False.

13. Code the following

    Screen Capture #10 (1 point)



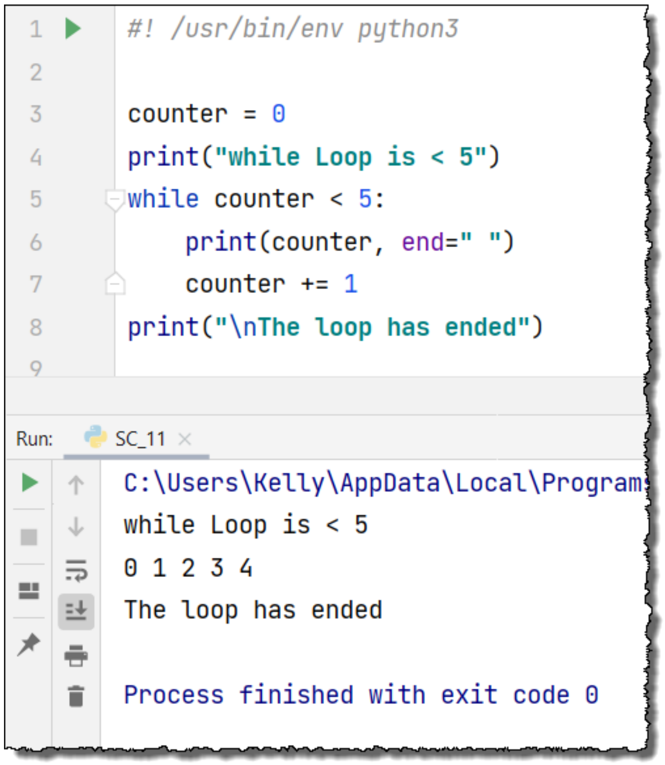
```python
#! /usr/bin/env python3

choice = "y"
while choice.lower() == "y":
    print("Hello!")
    choice = input("Say Hello again? (y/n): ")

print("Good bye!")
```

Run test

```
C:\Users\Saddleback\AppData\Local\Programs\Python\
Hello!
Say Hello again? (y/n): y
Hello!
Say Hello again? (y/n): y
Hello!
Say Hello again? (y/n): n
Good bye!

Process finished with exit code 0
```

14. Code the following

Screen Capture #11 (1 point)



```
#! /usr/bin/env python3

counter = 0
print("while Loop is < 5")
while counter < 5:
    print(counter, end=" ")
    counter += 1
print("\nThe loop has ended")
```

Run: SC_11

```
C:\Users\Kelly\AppData\Local\Program
while Loop is < 5
0 1 2 3 4
The loop has ended

Process finished with exit code 0
```

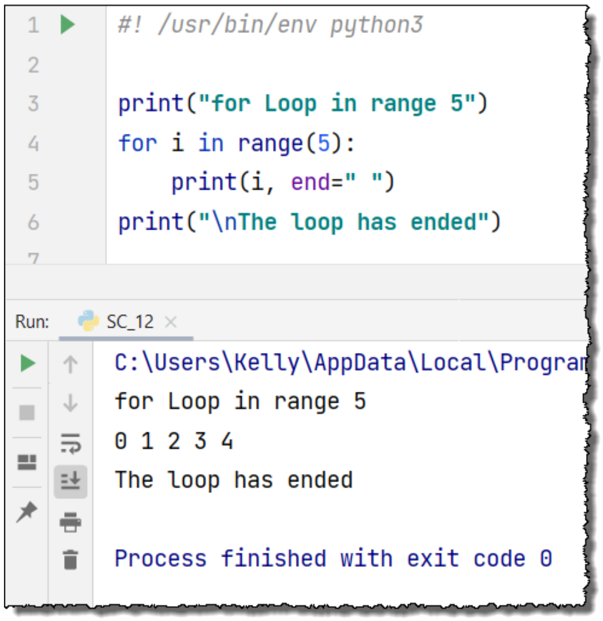
## range()

| Function | |
|---|---|
| **range(stop)** | Returns integer values from 0 to stop |
| **range(start, stop [, step])** | Returns integer values from the start to the stop with optional step value |

### *range(stop)*

15. Code the following

Screen Capture #12 (1 point)



```
#! /usr/bin/env python3

print("for Loop in range 5")
for i in range(5):
    print(i, end=" ")
print("\nThe loop has ended")
```

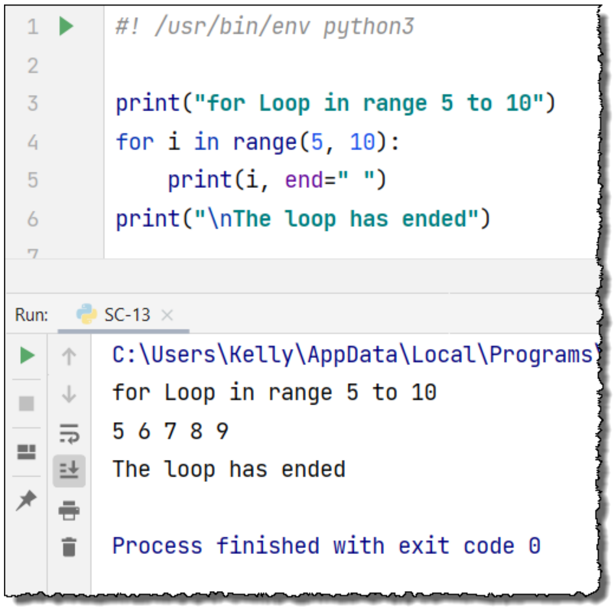
Run: SC_12

```
C:\Users\Kelly\AppData\Local\Progran
for Loop in range 5
0 1 2 3 4
The loop has ended

Process finished with exit code 0
```

### *range(start, stop)*

16. Code the following

Screen Capture #13 (1 point)



```
#! /usr/bin/env python3

print("for Loop in range 5 to 10")
for i in range(5, 10):
    print(i, end=" ")
print("\nThe loop has ended")
```

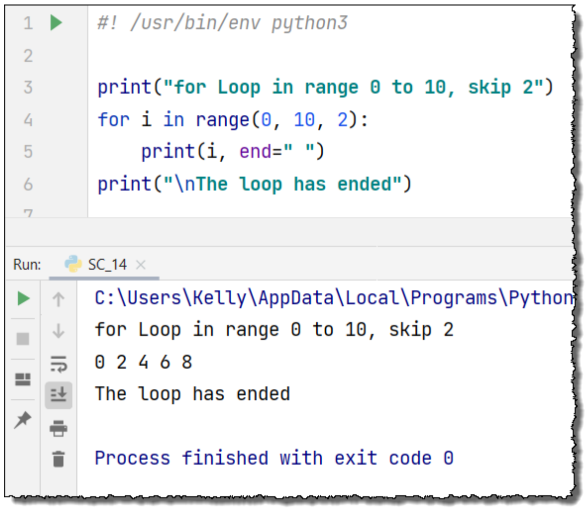
Run: SC-13

```
C:\Users\Kelly\AppData\Local\Programs\
for Loop in range 5 to 10
5 6 7 8 9
The loop has ended

Process finished with exit code 0
```

*range(start, stop, step)*

17. Code the following

Screen Capture #14 (1 point)



```
#! /usr/bin/env python3

print("for Loop in range 0 to 10, skip 2")
for i in range(0, 10, 2):
    print(i, end=" ")
print("\nThe loop has ended")
```

Run: SC_14

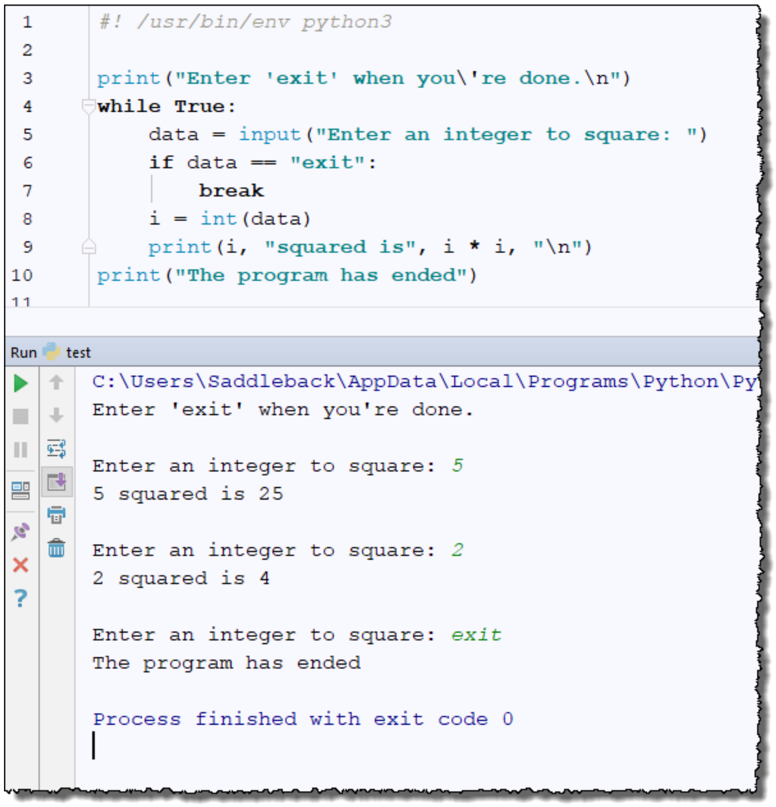
```
C:\Users\Kelly\AppData\Local\Programs\Python
for Loop in range 0 to 10, skip 2
0 2 4 6 8
The loop has ended

Process finished with exit code 0
```

*break*

The break statement allows you to jump out of the loop and execute the next statement following the loop's final statement.

18. Code the following

Screen Capture #15 (1 point)



```
#! /usr/bin/env python3

print("Enter 'exit' when you\'re done.\n")
while True:
    data = input("Enter an integer to square: ")
    if data == "exit":
        break
    i = int(data)
    print(i, "squared is", i * i, "\n")
print("The program has ended")
```

Run test

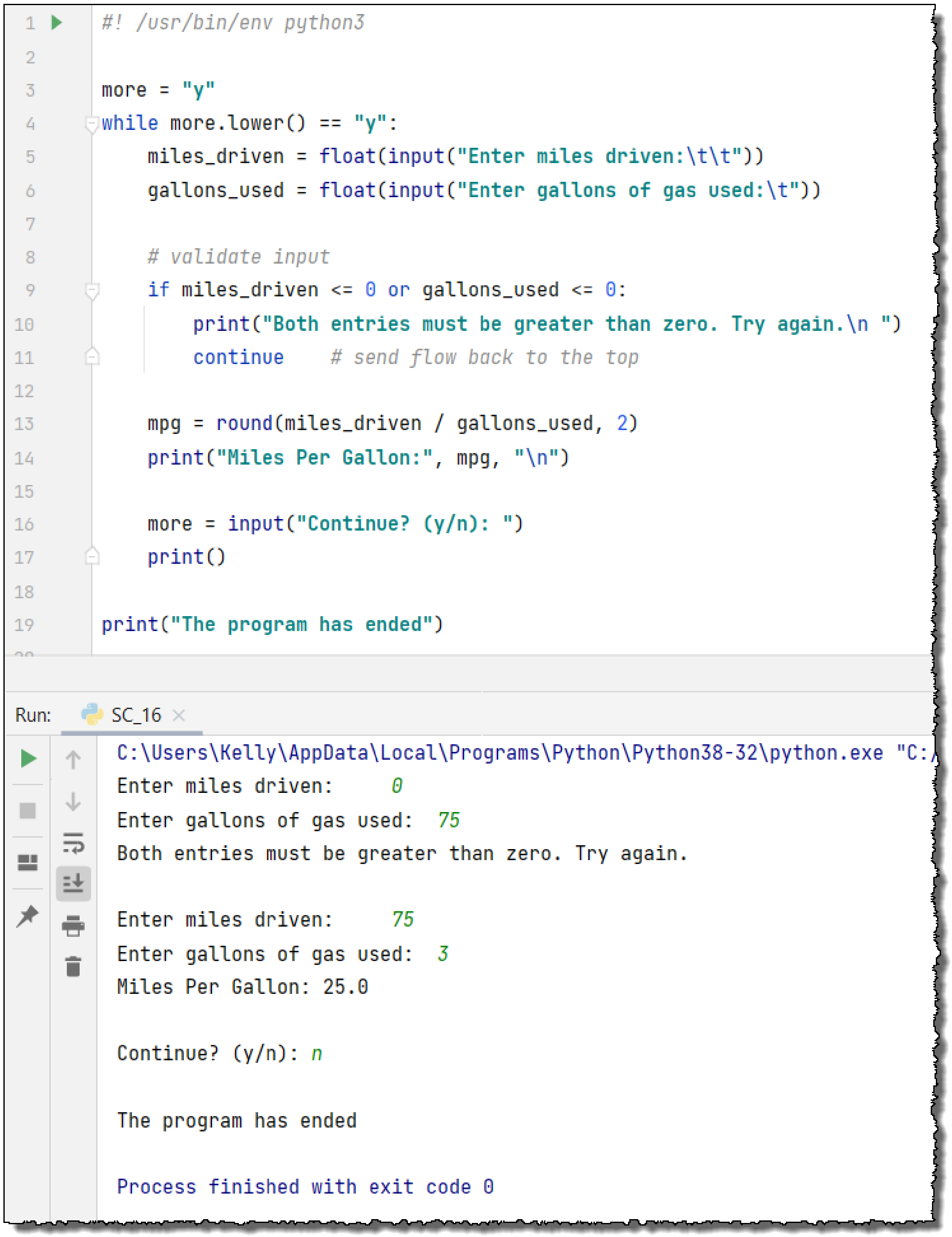
```
C:\Users\Saddleback\AppData\Local\Programs\Python\Py
Enter 'exit' when you're done.

Enter an integer to square: 5
5 squared is 25

Enter an integer to square: 2
2 squared is 4

Enter an integer to square: exit
The program has ended

Process finished with exit code 0
```

*continue*

The continue statement allows you to jump back to the top of the loop for the next iteration.

19. Code the following

    Screen Capture #16 (2 point)



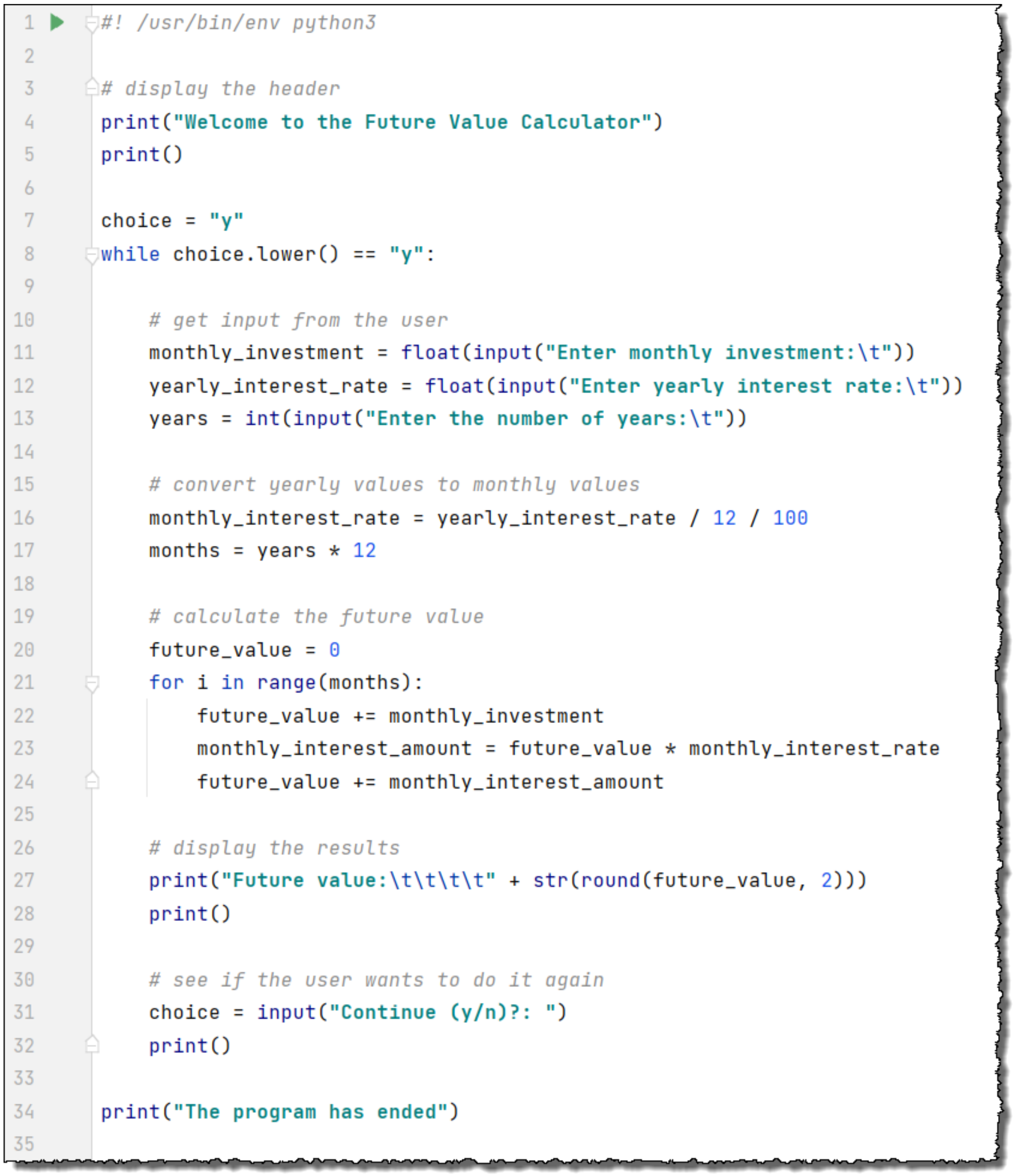
```python
#! /usr/bin/env python3

more = "y"
while more.lower() == "y":
    miles_driven = float(input("Enter miles driven:\t\t"))
    gallons_used = float(input("Enter gallons of gas used:\t"))

    # validate input
    if miles_driven <= 0 or gallons_used <= 0:
        print("Both entries must be greater than zero. Try again.\n ")
        continue    # send flow back to the top

    mpg = round(miles_driven / gallons_used, 2)
    print("Miles Per Gallon:", mpg, "\n")

    more = input("Continue? (y/n): ")
    print()

print("The program has ended")
```

Run: SC_16

```
C:\Users\Kelly\AppData\Local\Programs\Python\Python38-32\python.exe "C:/
Enter miles driven:     0
Enter gallons of gas used:   75
Both entries must be greater than zero. Try again.

Enter miles driven:     75
Enter gallons of gas used:   3
Miles Per Gallon: 25.0

Continue? (y/n): n

The program has ended

Process finished with exit code 0
```

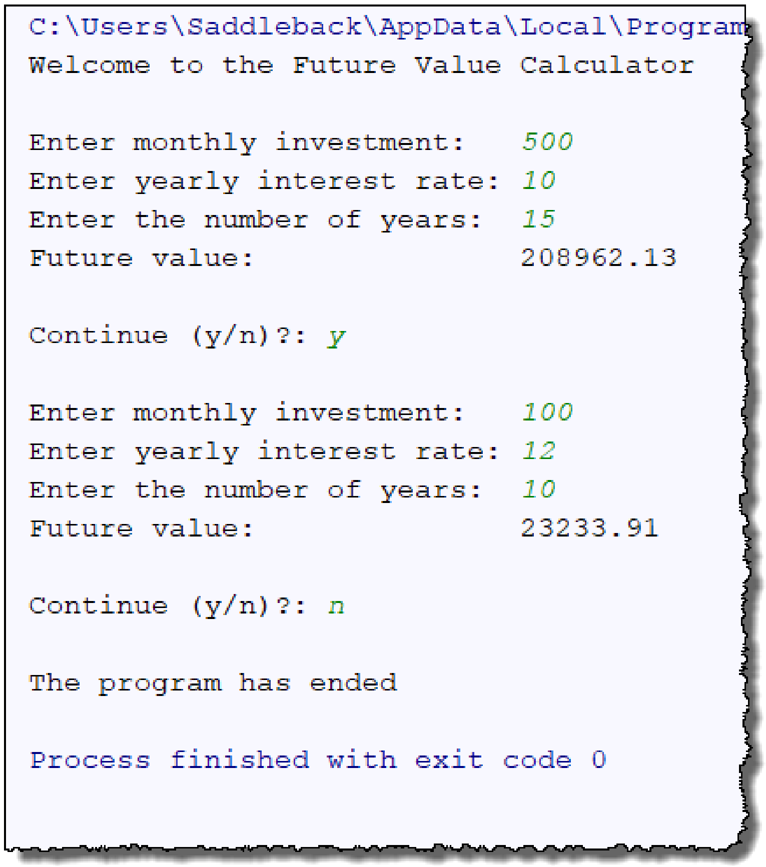
## Future Value Program

20. Create a python file named **future_value.py**. Make sure you attach this file when submitting the assignment.
21. Code the following



```python
#! /usr/bin/env python3

# display the header
print("Welcome to the Future Value Calculator")
print()

choice = "y"
while choice.lower() == "y":

    # get input from the user
    monthly_investment = float(input("Enter monthly investment:\t"))
    yearly_interest_rate = float(input("Enter yearly interest rate:\t"))
    years = int(input("Enter the number of years:\t"))

    # convert yearly values to monthly values
    monthly_interest_rate = yearly_interest_rate / 12 / 100
    months = years * 12

    # calculate the future value
    future_value = 0
    for i in range(months):
        future_value += monthly_investment
        monthly_interest_amount = future_value * monthly_interest_rate
        future_value += monthly_interest_amount

    # display the results
    print("Future value:\t\t\t\t" + str(round(future_value, 2)))
    print()

    # see if the user wants to do it again
    choice = input("Continue (y/n)?: ")
    print()

print("The program has ended")

```

22. Test using the same values:
    Screen Capture #17 (2 points)
    Code Validation – future_value.py (1 points)



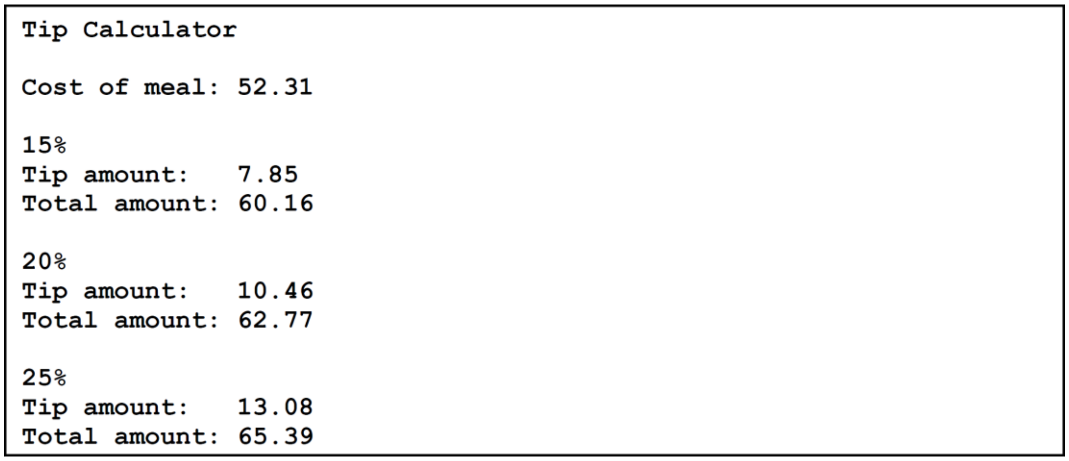
```
C:\Users\Saddleback\AppData\Local\Program
Welcome to the Future Value Calculator

Enter monthly investment:    500
Enter yearly interest rate: 10
Enter the number of years:  15
Future value:               208962.13

Continue (y/n)?: y

Enter monthly investment:    100
Enter yearly interest rate: 12
Enter the number of years:  10
Future value:               23233.91

Continue (y/n)?: n

The program has ended

Process finished with exit code 0
```

## Extra Credit

To get full points for each extra credit, you must include screen captures of the running output as well as the python (.py) code files.

### Extra Credit #1 – Tip Calculator (+1 Extra Credit)

Create a program that calculates three options for an appropriate tip to leave after a meal at a restaurant.



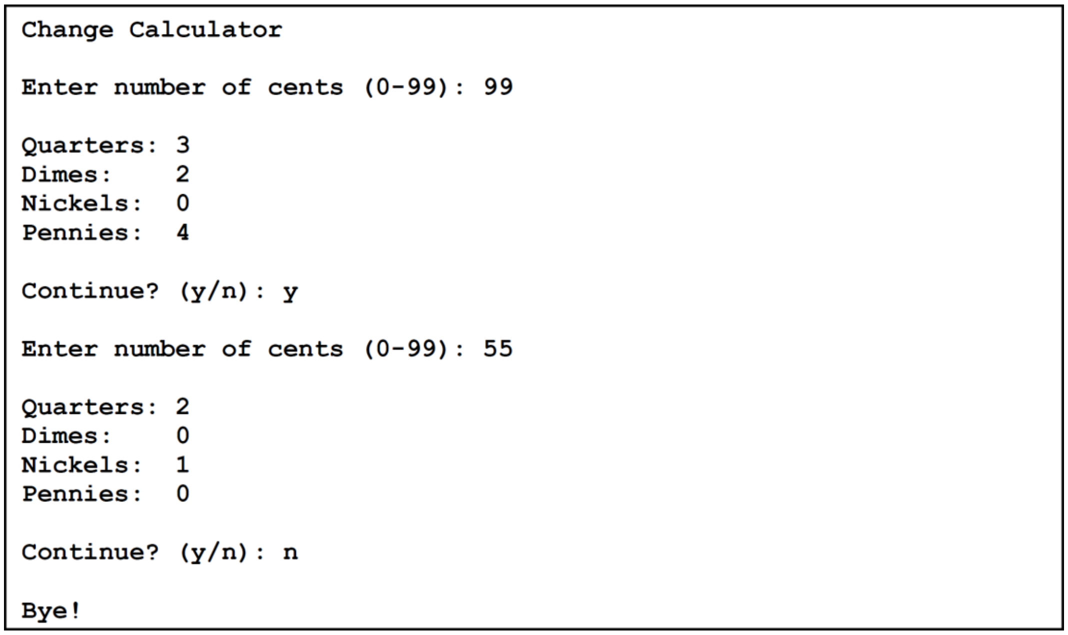
```
Tip Calculator

Cost of meal: 52.31

15%
Tip amount:   7.85
Total amount: 60.16

20%
Tip amount:   10.46
Total amount: 62.77

25%
Tip amount:   13.08
Total amount: 65.39
```

*Specifications:*

- The program should calculate and display the cost of tipping at 15%, 20%, and 25% tips.
- Assume the user will enter valid data.
- The program should round results to a maximum of two decimal places.

## Extra Credit #2 – Change Calculator (+1 Extra Credit)

Create a program that calculates the coins needed to make changes for the specified.



```
Change Calculator

Enter number of cents (0-99): 99

Quarters: 3
Dimes:    2
Nickels:  0
Pennies:  4

Continue? (y/n): y

Enter number of cents (0-99): 55

Quarters: 2
Dimes:    0
Nickels:  1
Pennies:  0

Continue? (y/n): n

Bye!
```

*Specifications:*

- The program should display the maximum number of quarters, dimes, nickels, and pennies that one needs to make up the specified number of cents.
- Assume the user will enter a valid integer for the number of cents.
- The program should continue only if the user enters “y” or “Y” to continue/