Assignment 1: Getting Started

Milan Bista

University of Cumberlands

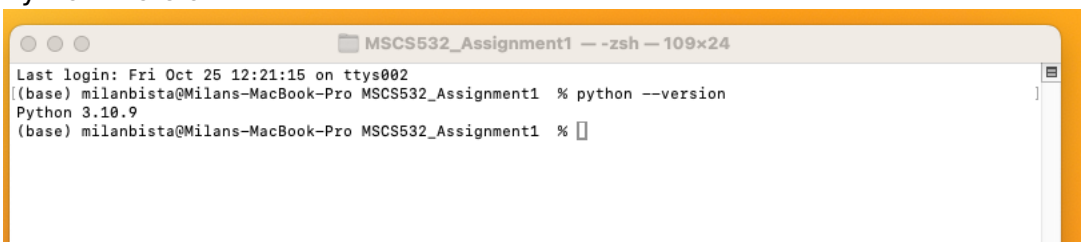
2024 Fall - Algorithms and Data Structures (MSCS-532-B01) - Second Bi-term

***Instructors: Machica Mcclain /* Vanessa Cooper**

GitHub: https://github.com/mbista25742/MSCS532_Assignment1

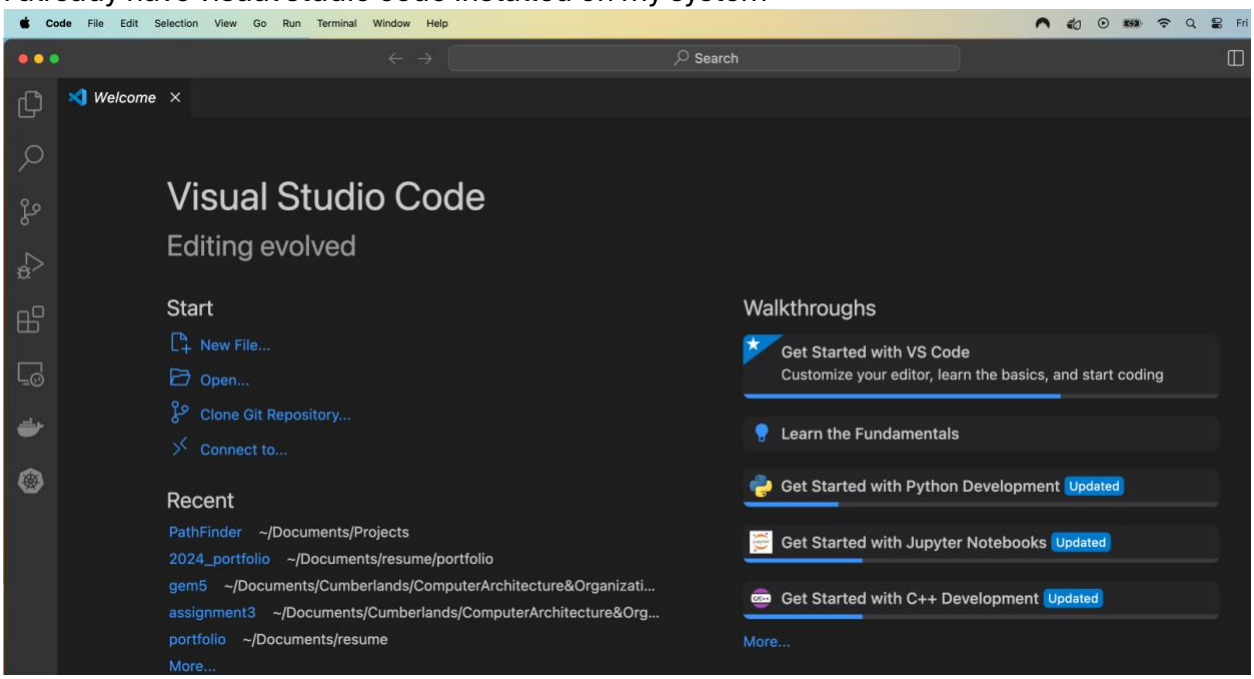
Step 1: Install Python

I checked the python installed on my computer with following command
Python –version



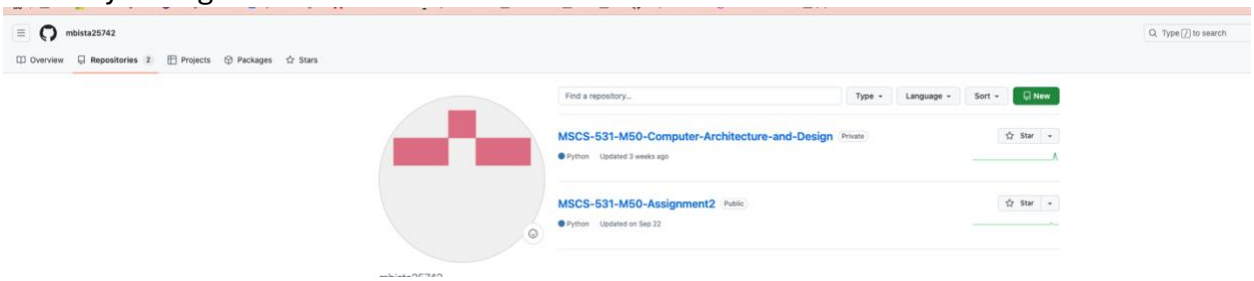
Step 2: Install Visual Studio Code (VS Code) or an IDE of choice

I already have visual studio code installed on my system



Step 3: GitHub Account Creation

I already have github account created from other classes



Step 4: Assignment

Now I am going to do the first assignment which is to write an insertion sort algorithm to sort in monotonically decreasing order
I have created a repository named **MSCS532_Assignment1**

GithubLink: https://github.com/mbista25742/MSCS532_Assignment1

Insertion Sort is a simple and intuitive sorting algorithm that builds a sorted array (or list) one element at a time by repeatedly taking the next unsorted element and inserting it into its correct position within the sorted portion of the array. It is particularly efficient for small datasets or partially sorted data.

Following is the python script to sort an array using insertion sort in monotonically decreasing order:

token | InsertionSort.py M | Assignment 1 Getting Sta

MSCS532_Assignment1 > InsertionSort.py > ...

```python
arr = [5,2,9,1,7, 0,4,2,1,8,23,45,34]

for i in range(1, len(arr)):
    current = arr[i]
    j = i-1
    while j >=0 and arr[j] < current:
        arr[j+1] = arr[j]
        j = j-1
    arr[j+1] = current

print(arr)
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

Output is:

token | InsertionSort.py M | Assignment 1 Getting Started.docx

MSCS532_Assignment1 > InsertionSort.py > ...

```python
arr = [5,2,9,1,7, 0,4,2,1,8,23,45,34]

for i in range(1, len(arr)):
    current = arr[i]
    j = i-1
    while j >=0 and arr[j] < current:
        arr[j+1] = arr[j]
        j = j-1
    arr[j+1] = current

print(arr)
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS zsh - MS

```
(base) milanbista@Milans-MacBook-Pro MSCS532_Assignment1 % python InsertionSort.py
[45, 34, 23, 9, 8, 7, 5, 4, 2, 2, 1, 1, 0]
(base) milanbista@Milans-MacBook-Pro MSCS532_Assignment1 %
```

Insertion Sort has a time complexity that varies based on the input arrangement. In the best case, when the array is already sorted, it operates in linear time, O(n), requiring only a single comparison for each element. In contrast, the average and worst-case time complexity is O(n^2) due to the need to compare and shift previously sorted elements. Despite this, Insertion Sort is efficient for small or nearly sorted arrays. Its space complexity is O(1), as it sorts the array in place without requiring additional memory, making it a suitable choice when memory usage is a concern.