Software Engineering - IT 314
Lab-05

Name : Abhimanyu Negi
ID: 202001080

Steps for Static Analysis

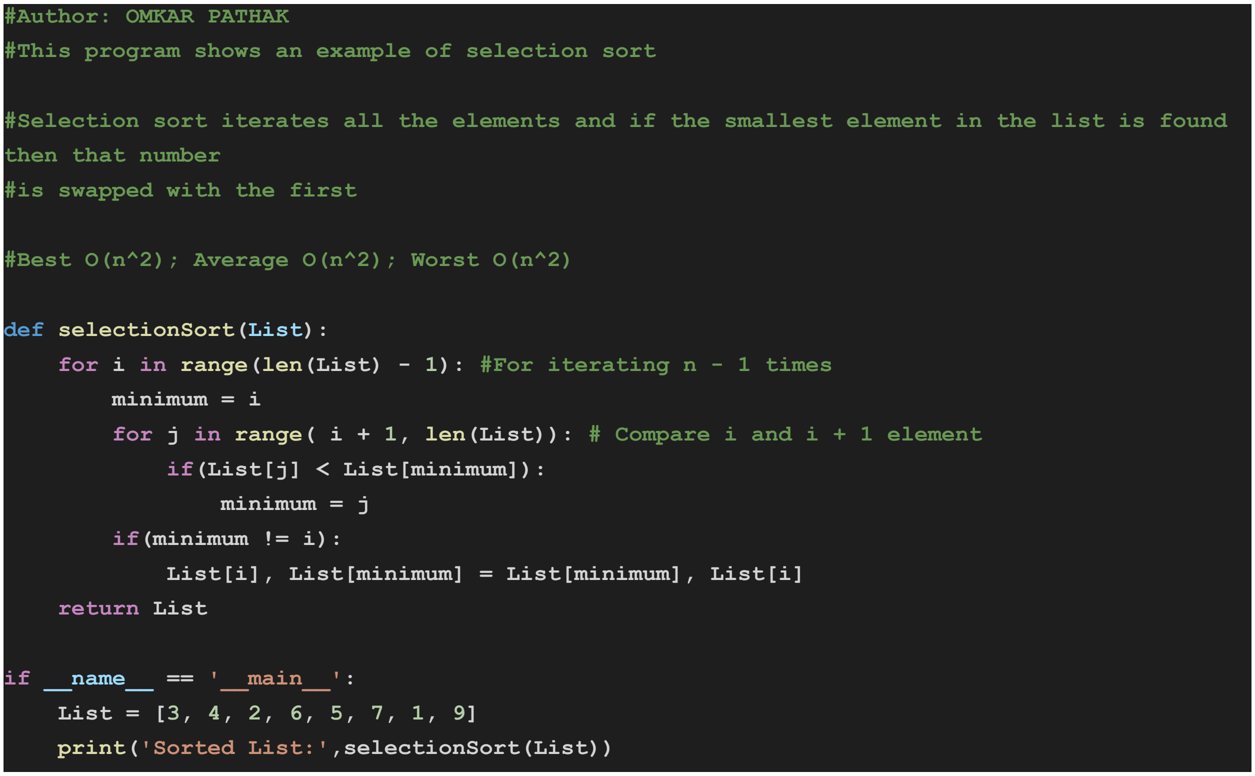
For the exercise we are using PyLint for performing the static analysis:

Installation & Analysis

1. First run the command prompt
2. Enter the command “ pip install pylint”
3. After successful installation of pylint, change the directory to the folder with all the files which have to be static analyzed.
4. Run the files with the help of command prompt by entering the command “py -m pylint filename.py”

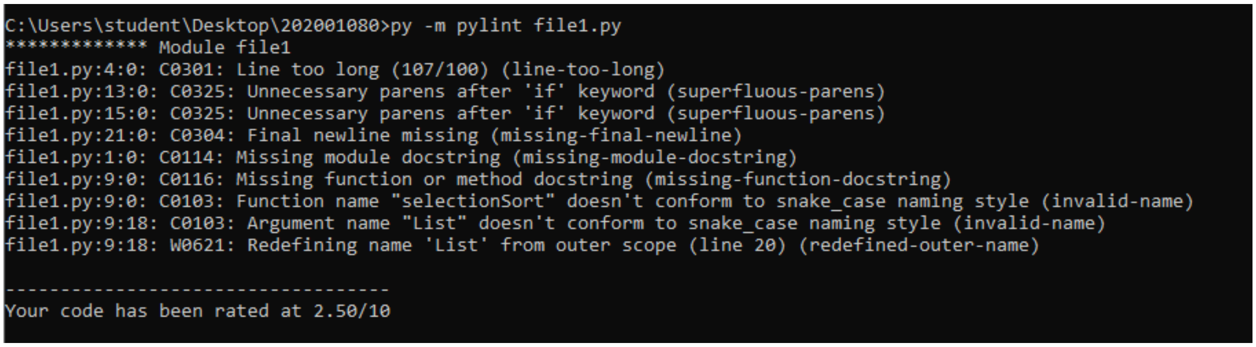
Code and Output

File 1 - Code



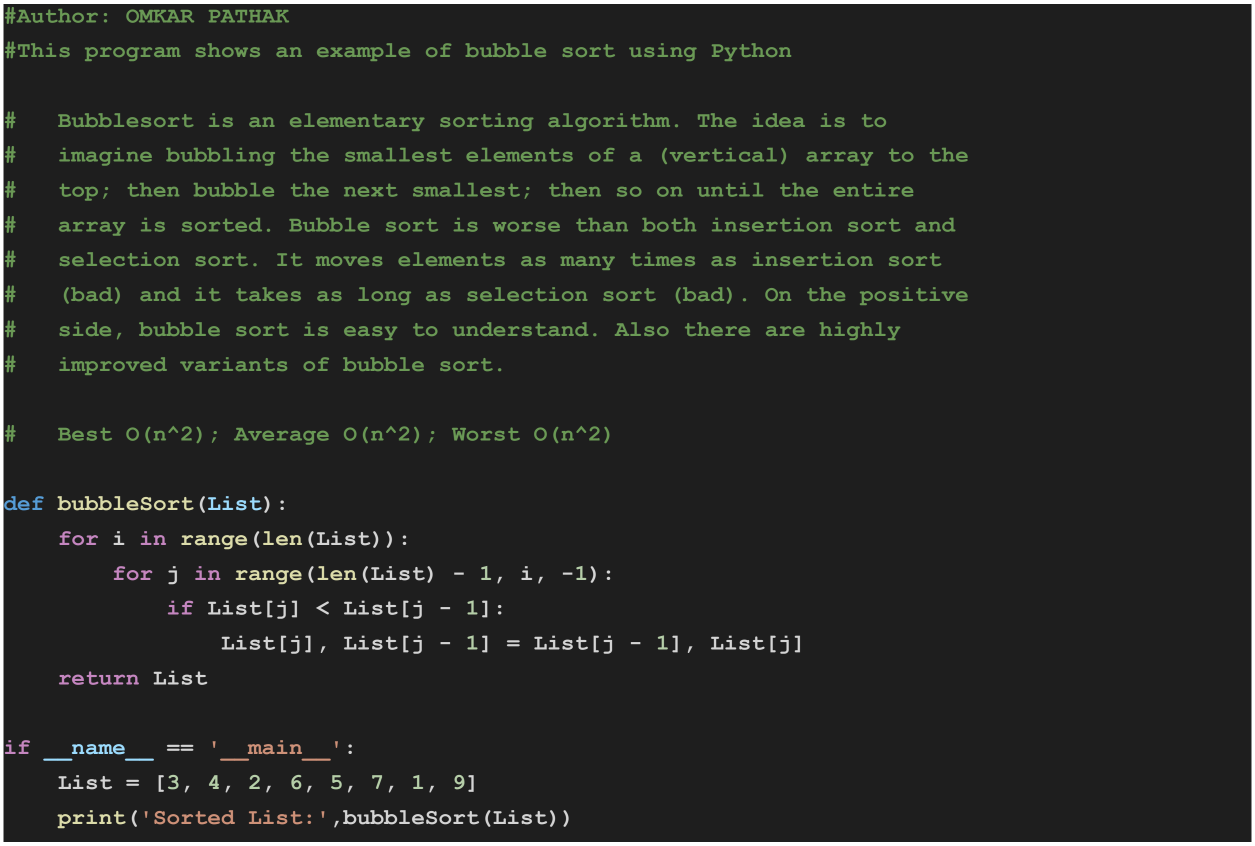
```python
#Author: OMKAR PATHAK
#This program shows an example of selection sort

#Selection sort iterates all the elements and if the smallest element in the list is found then that number
#is swapped with the first

#Best O(n^2); Average O(n^2); Worst O(n^2)

def selectionSort(List):
    for i in range(len(List) - 1): #For iterating n - 1 times
        minimum = i
        for j in range( i + 1, len(List)): # Compare i and i + 1 element
            if(List[j] < List[minimum]):
                minimum = j
        if(minimum != i):
            List[i], List[minimum] = List[minimum], List[i]
    return List

if __name__ == '__main__':
    List = [3, 4, 2, 6, 5, 7, 1, 9]
    print('Sorted List:',selectionSort(List))
```

Output:



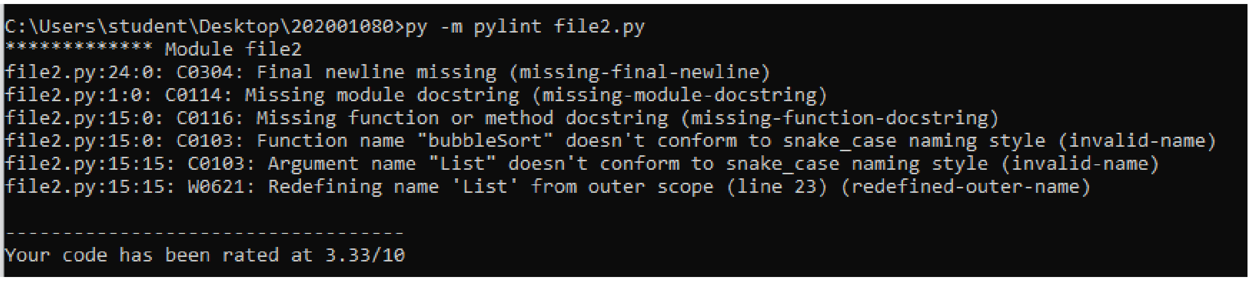
```
C:\Users\student\Desktop\202001080>py -m pylint file1.py
************* Module file1
file1.py:4:0: C0301: Line too long (107/100) (line-too-long)
file1.py:13:0: C0325: Unnecessary parens after 'if' keyword (superfluous-parens)
file1.py:15:0: C0325: Unnecessary parens after 'if' keyword (superfluous-parens)
file1.py:21:0: C0304: Final newline missing (missing-final-newline)
file1.py:1:0: C0114: Missing module docstring (missing-module-docstring)
file1.py:9:0: C0116: Missing function or method docstring (missing-function-docstring)
file1.py:9:0: C0103: Function name "selectionSort" doesn't conform to snake_case naming style (invalid-name)
file1.py:9:18: C0103: Argument name "List" doesn't conform to snake_case naming style (invalid-name)
file1.py:9:18: W0621: Redefining name 'List' from outer scope (line 20) (redefined-outer-name)

-----------------------------------
Your code has been rated at 2.50/10
```

File2 - Code



```
#Author: OMKAR PATHAK
#This program shows an example of bubble sort using Python

#   Bubblesort is an elementary sorting algorithm. The idea is to
#   imagine bubbling the smallest elements of a (vertical) array to the
#   top; then bubble the next smallest; then so on until the entire
#   array is sorted. Bubble sort is worse than both insertion sort and
#   selection sort. It moves elements as many times as insertion sort
#   (bad) and it takes as long as selection sort (bad). On the positive
#   side, bubble sort is easy to understand. Also there are highly
#   improved variants of bubble sort.

#   Best O(n^2); Average O(n^2); Worst O(n^2)

def bubbleSort(List):
    for i in range(len(List)):
        for j in range(len(List) - 1, i, -1):
            if List[j] < List[j - 1]:
                List[j], List[j - 1] = List[j - 1], List[j]
    return List

if __name__ == '__main__':
    List = [3, 4, 2, 6, 5, 7, 1, 9]
    print('Sorted List:',bubbleSort(List))
```

Output:

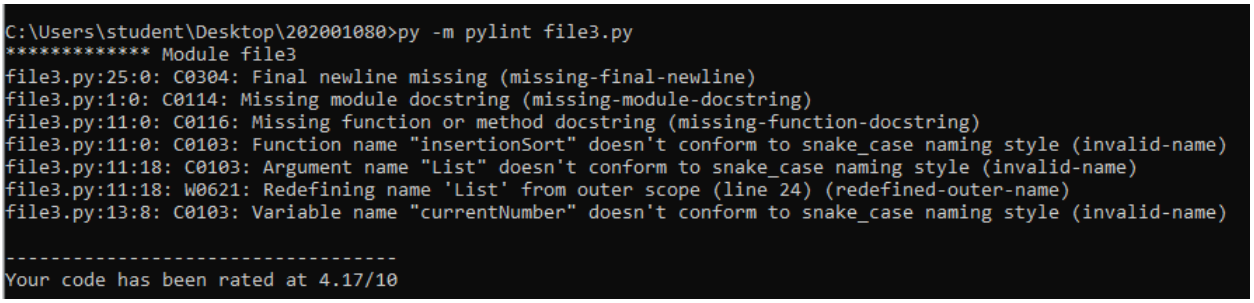


```
C:\Users\student\Desktop\202001080>py -m pylint file2.py
************* Module file2
file2.py:24:0: C0304: Final newline missing (missing-final-newline)
file2.py:1:0: C0114: Missing module docstring (missing-module-docstring)
file2.py:15:0: C0116: Missing function or method docstring (missing-function-docstring)
file2.py:15:0: C0103: Function name "bubbleSort" doesn't conform to snake_case naming style (invalid-name)
file2.py:15:15: C0103: Argument name "List" doesn't conform to snake_case naming style (invalid-name)
file2.py:15:15: W0621: Redefining name 'List' from outer scope (line 23) (redefined-outer-name)

-----------------------------------
Your code has been rated at 3.33/10
```

File3: Code

```
#Author: OMKAR PATHAK
#This program shows an example of insertion sort using Python

#  Insertion sort is good for collections that are very small
#  or nearly sorted. Otherwise it's not a good sorting algorithm:
#  it moves data around too much. Each time an insertion is made,
#  all elements in a greater position are shifted.

#  Best O(n); Average O(n^2); Worst O(n^2)

def insertionSort(List):
    for i in range(1, len(List)):
        currentNumber = List[i]
        for j in range(i - 1, -1, -1):
            if List[j] > currentNumber :
                List[j], List[j + 1] = List[j + 1], List[j]
            else:
                List[j + 1] = currentNumber
                break

    return List

if __name__ == '__main__':
    List = [3, 4, 2, 6, 5, 7, 1, 9]
    print('Sorted List:',insertionSort(List))
```

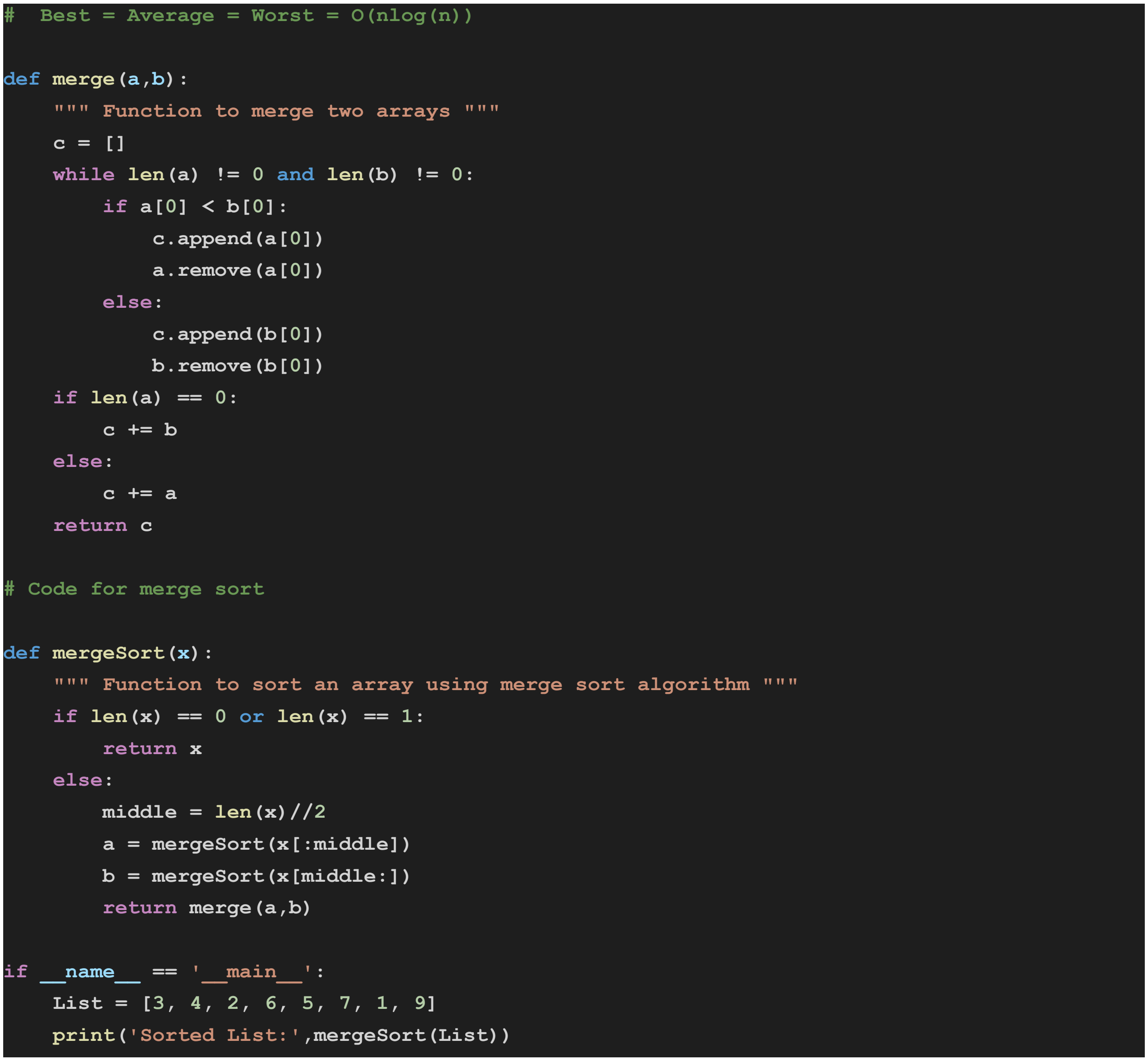
Output:



```
C:\Users\student\Desktop\202001080>py -m pylint file3.py
************* Module file3
file3.py:25:0: C0304: Final newline missing (missing-final-newline)
file3.py:1:0: C0114: Missing module docstring (missing-module-docstring)
file3.py:11:0: C0116: Missing function or method docstring (missing-function-docstring)
file3.py:11:0: C0103: Function name "insertionSort" doesn't conform to snake_case naming style (invalid-name)
file3.py:11:18: C0103: Argument name "List" doesn't conform to snake_case naming style (invalid-name)
file3.py:11:18: W0621: Redefining name 'List' from outer scope (line 24) (redefined-outer-name)
file3.py:13:8: C0103: Variable name "currentNumber" doesn't conform to snake_case naming style (invalid-name)

-----------------------------------
Your code has been rated at 4.17/10
```

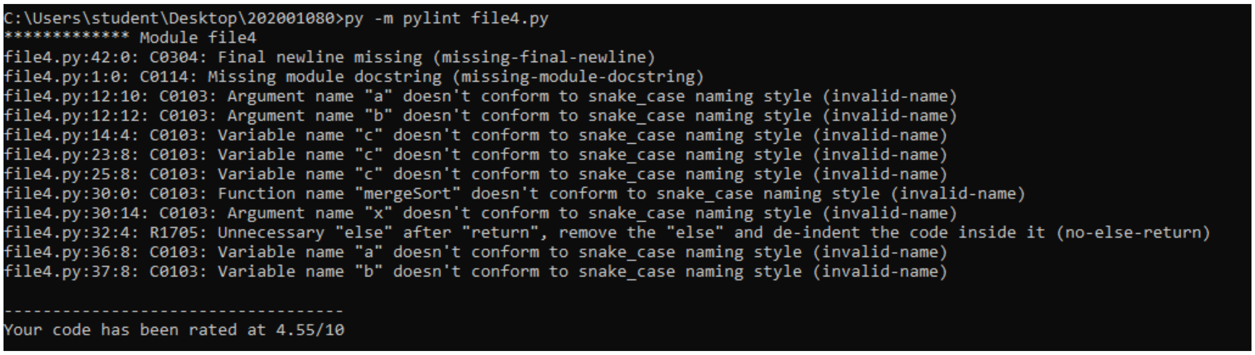
File 4: Code

```
#Author: OMKAR PATHAK
#This program gives an example of Merge sort

#   Merge sort is a divide and conquer algorithm. In the divide and
#   conquer paradigm, a problem is broken into pieces where each piece
#   still retains all the properties of the larger problem -- except
#   its size. To solve the original problem, each piece is solved
#   individually; then the pieces are merged back together.
```



```python
#  Best = Average = Worst = O(nlog(n))

def merge(a,b):
    """ Function to merge two arrays """
    c = []
    while len(a) != 0 and len(b) != 0:
        if a[0] < b[0]:
            c.append(a[0])
            a.remove(a[0])
        else:
            c.append(b[0])
            b.remove(b[0])
    if len(a) == 0:
        c += b
    else:
        c += a
    return c

# Code for merge sort

def mergeSort(x):
    """ Function to sort an array using merge sort algorithm """
    if len(x) == 0 or len(x) == 1:
        return x
    else:
        middle = len(x)//2
        a = mergeSort(x[:middle])
        b = mergeSort(x[middle:])
        return merge(a,b)

if __name__ == '__main__':
    List = [3, 4, 2, 6, 5, 7, 1, 9]
    print('Sorted List:',mergeSort(List))
```

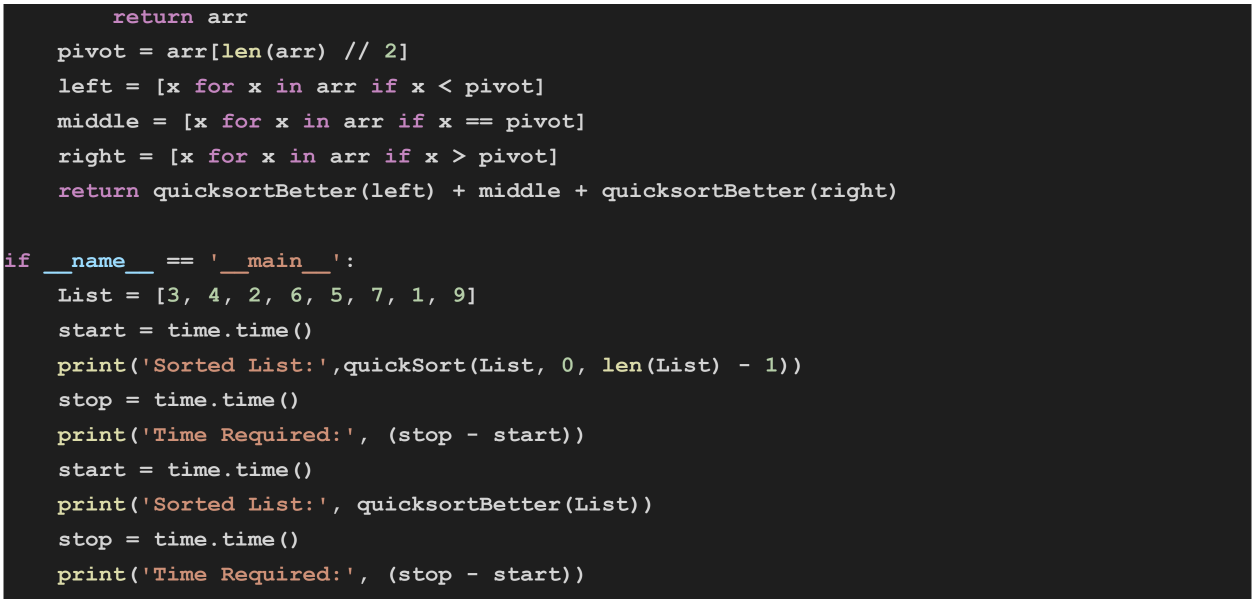
Output:



```
C:\Users\student\Desktop\202001080>py -m pylint file4.py
************* Module file4
file4.py:42:0: C0304: Final newline missing (missing-final-newline)
file4.py:1:0: C0114: Missing module docstring (missing-module-docstring)
file4.py:12:10: C0103: Argument name "a" doesn't conform to snake_case naming style (invalid-name)
file4.py:12:12: C0103: Argument name "b" doesn't conform to snake_case naming style (invalid-name)
file4.py:14:4: C0103: Variable name "c" doesn't conform to snake_case naming style (invalid-name)
file4.py:23:8: C0103: Variable name "c" doesn't conform to snake_case naming style (invalid-name)
file4.py:25:8: C0103: Variable name "c" doesn't conform to snake_case naming style (invalid-name)
file4.py:30:0: C0103: Function name "mergeSort" doesn't conform to snake_case naming style (invalid-name)
file4.py:30:14: C0103: Argument name "x" doesn't conform to snake_case naming style (invalid-name)
file4.py:32:4: R1705: Unnecessary "else" after "return", remove the "else" and de-indent the code inside it (no-else-return)
file4.py:36:8: C0103: Variable name "a" doesn't conform to snake_case naming style (invalid-name)
file4.py:37:8: C0103: Variable name "b" doesn't conform to snake_case naming style (invalid-name)

-----------------------------------
Your code has been rated at 4.55/10
```

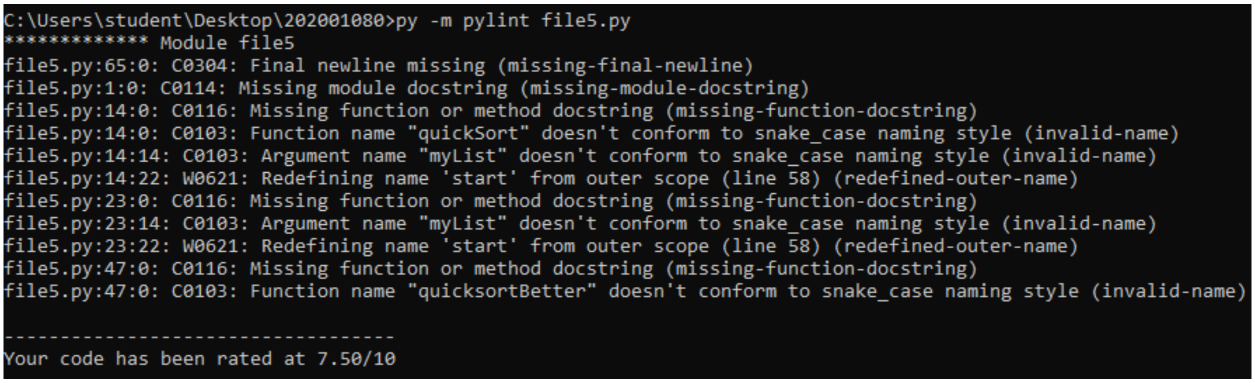
File 5: Code

```
#Author: OMKAR PATHAK
#This program illustrates an example of quick sort

#  Quicksort works by selecting an element called a pivot and splitting
#  the array around that pivot such that all the elements in, say, the
#  left sub-array are less than pivot and all the elements in the right
#  sub-array are greater than pivot. The splitting continues until the
#  array can no longer be broken into pieces. That's it. Quicksort is
#  done.

#  Best = Average = O(nlog(n)); Worst = O(n^2
import time

def quickSort(myList, start, end):
    if start < end:
        # partition the list
        pivot = partition(myList, start, end)
        # sort both halves
        quickSort(myList, start, pivot-1)
        quickSort(myList, pivot+1, end)
    return myList

def partition(myList, start, end):
    pivot = myList[start]
    left = start+1
    right = end
    done = False
    while not done:
        while left <= right and myList[left] <= pivot:
            left = left + 1
        while myList[right] >= pivot and right >=left:
            right = right -1
        if right < left:
            done= True
        else:
            # swap places
            temp=myList[left]
            myList[left]=myList[right]
            myList[right]=temp
    # swap start with myList[right]
    temp=myList[start]
    myList[start]=myList[right]
    myList[right]=temp
    return right

# A more efficient solution
def quicksortBetter(arr):
    if len(arr) <= 1:
```



```
        return arr
    pivot = arr[len(arr) // 2]
    left = [x for x in arr if x < pivot]
    middle = [x for x in arr if x == pivot]
    right = [x for x in arr if x > pivot]
    return quicksortBetter(left) + middle + quicksortBetter(right)

if __name__ == '__main__':
    List = [3, 4, 2, 6, 5, 7, 1, 9]
    start = time.time()
    print('Sorted List:',quickSort(List, 0, len(List) - 1))
    stop = time.time()
    print('Time Required:', (stop - start))
    start = time.time()
    print('Sorted List:', quicksortBetter(List))
    stop = time.time()
    print('Time Required:', (stop - start))
```

Output:



```
C:\Users\student\Desktop\202001080>py -m pylint file5.py
************* Module file5
file5.py:65:0: C0304: Final newline missing (missing-final-newline)
file5.py:1:0: C0114: Missing module docstring (missing-module-docstring)
file5.py:14:0: C0116: Missing function or method docstring (missing-function-docstring)
file5.py:14:0: C0103: Function name "quickSort" doesn't conform to snake_case naming style (invalid-name)
file5.py:14:14: C0103: Argument name "myList" doesn't conform to snake_case naming style (invalid-name)
file5.py:14:22: W0621: Redefining name 'start' from outer scope (line 58) (redefined-outer-name)
file5.py:23:0: C0116: Missing function or method docstring (missing-function-docstring)
file5.py:23:14: C0103: Argument name "myList" doesn't conform to snake_case naming style (invalid-name)
file5.py:23:22: W0621: Redefining name 'start' from outer scope (line 58) (redefined-outer-name)
file5.py:47:0: C0116: Missing function or method docstring (missing-function-docstring)
file5.py:47:0: C0103: Function name "quicksortBetter" doesn't conform to snake_case naming style (invalid-name)

-----------------------------------
Your code has been rated at 7.50/10
```

References
Codes: https://github.com/OmkarPathak/Python-Programs

Pylink :
https://pylint.readthedocs.io/en/latest/user_guide/messages/messages_overview.html