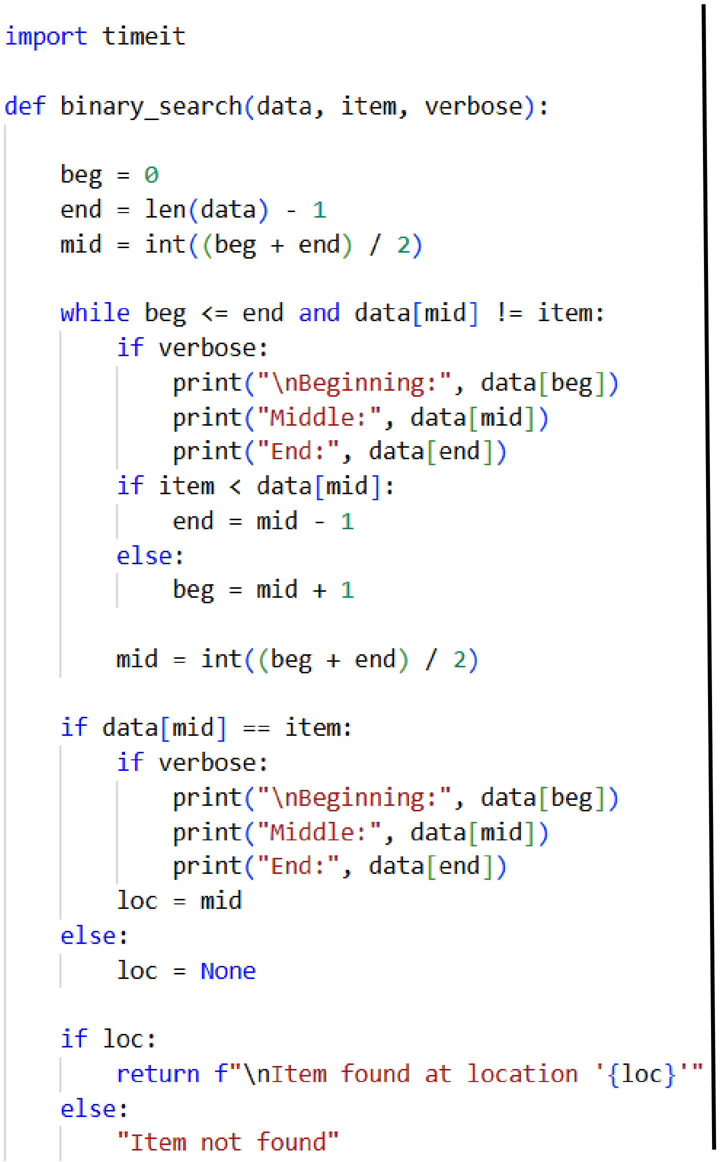
Usman Rasheed Siddiqui
NED University of Engineering and Technology

CIS Department
Data Structure and Algorithm

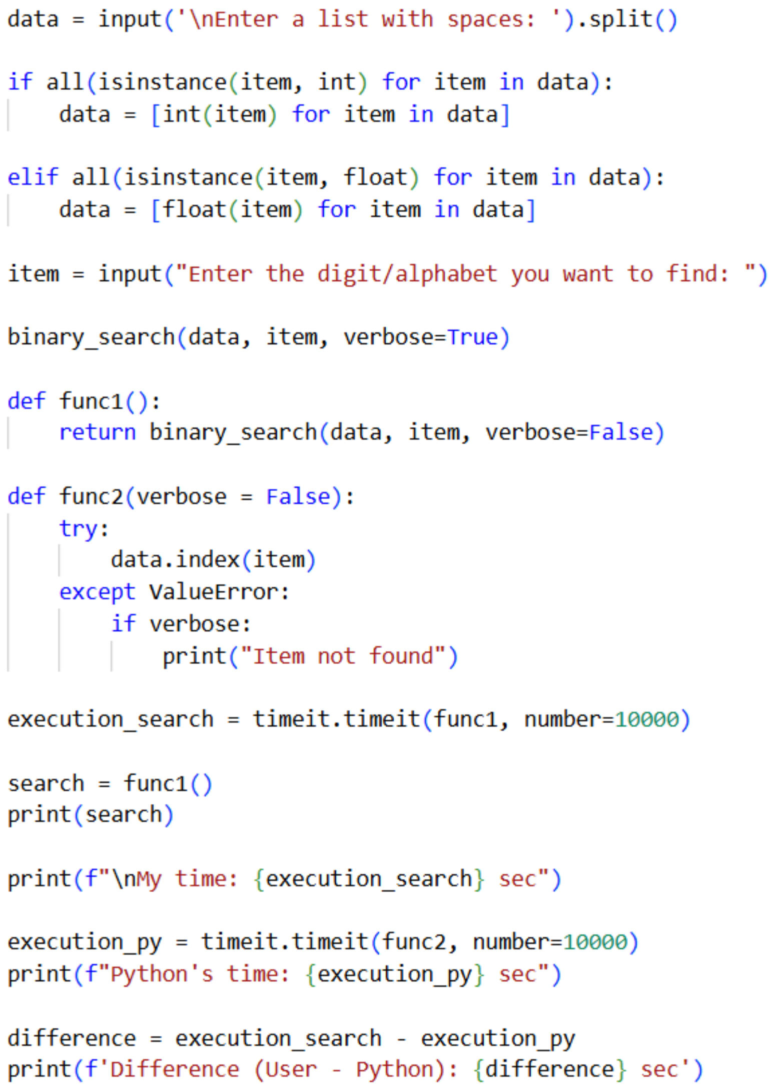
Lab Session 03

Exercise a:

CODE

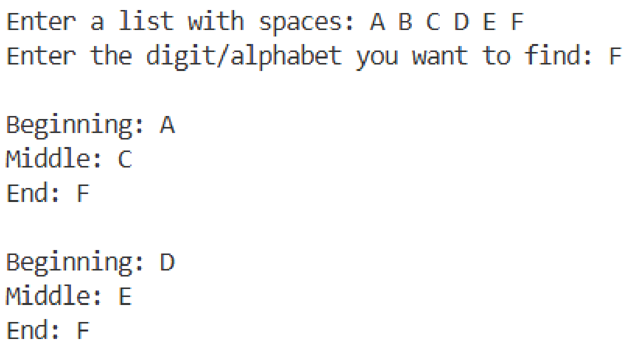


```python
import timeit

def binary_search(data, item, verbose):

    beg = 0
    end = len(data) - 1
    mid = int((beg + end) / 2)

    while beg <= end and data[mid] != item:
        if verbose:
            print("\nBeginning:", data[beg])
            print("Middle:", data[mid])
            print("End:", data[end])
        if item < data[mid]:
            end = mid - 1
        else:
            beg = mid + 1

        mid = int((beg + end) / 2)

    if data[mid] == item:
        if verbose:
            print("\nBeginning:", data[beg])
            print("Middle:", data[mid])
            print("End:", data[end])
        loc = mid
    else:
        loc = None

    if loc:
        return f"\nItem found at location '{loc}'"
    else:
        "Item not found"
```

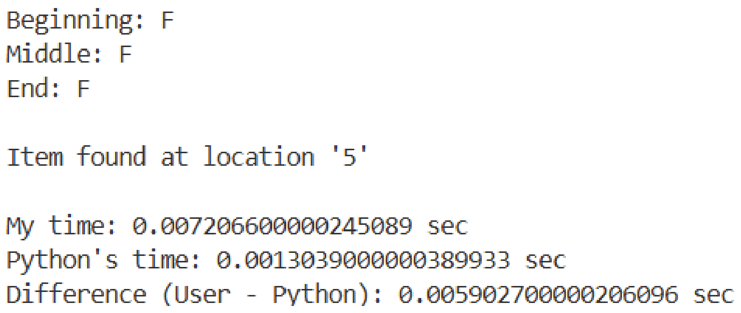


```python
data = input('\nEnter a list with spaces: ').split()

if all(isinstance(item, int) for item in data):
    data = [int(item) for item in data]

elif all(isinstance(item, float) for item in data):
    data = [float(item) for item in data]

item = input("Enter the digit/alphabet you want to find: ")

binary_search(data, item, verbose=True)

def func1():
    return binary_search(data, item, verbose=False)

def func2(verbose = False):
    try:
        data.index(item)
    except ValueError:
        if verbose:
            print("Item not found")

execution_search = timeit.timeit(func1, number=10000)

search = func1()
print(search)

print(f"\nMy time: {execution_search} sec")

execution_py = timeit.timeit(func2, number=10000)
print(f"Python's time: {execution_py} sec")

difference = execution_search - execution_py
print(f'Difference (User - Python): {difference} sec')
```

OUTPUT



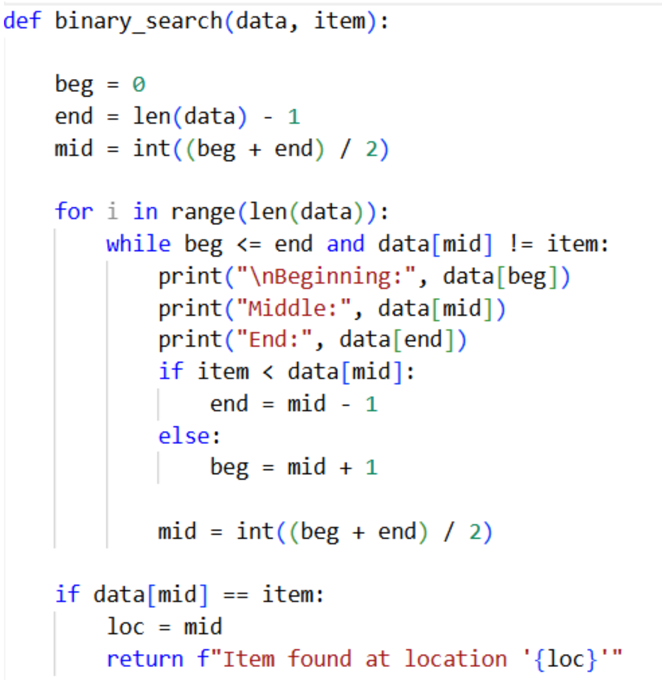
```
Enter a list with spaces: A B C D E F
Enter the digit/alphabet you want to find: F

Beginning: A
Middle: C
End: F

Beginning: D
Middle: E
End: F
```



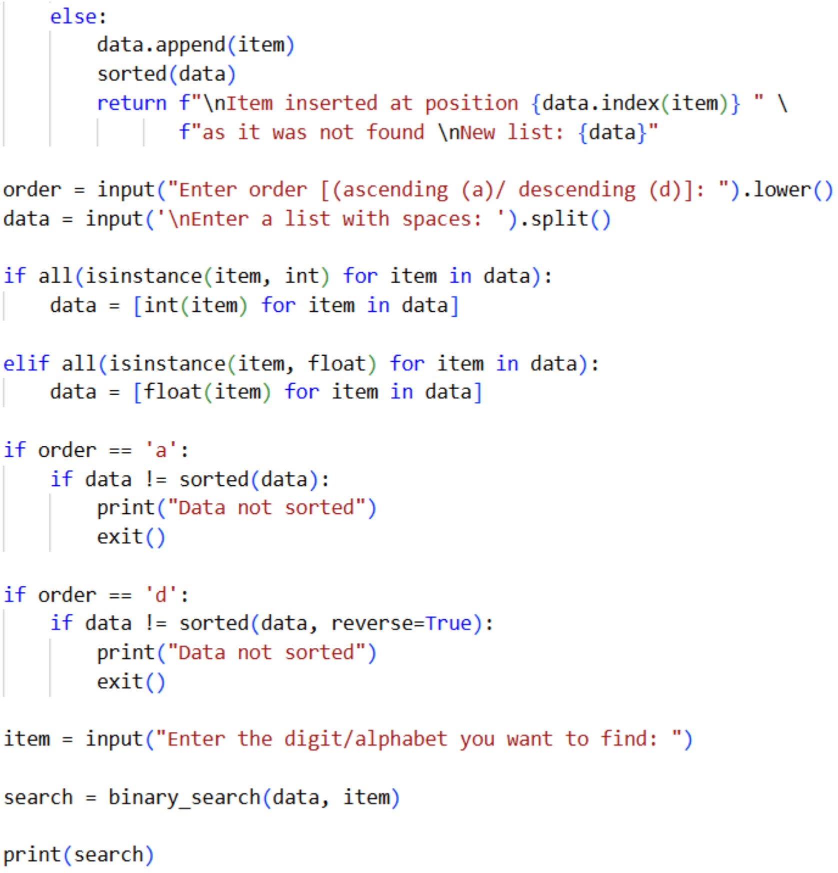
```
Beginning: F
Middle: F
End: F

Item found at location '5'

My time: 0.007206600000245089 sec
Python's time: 0.0013039000000389933 sec
Difference (User - Python): 0.005902700000206096 sec
```

Exercise c:

CODE

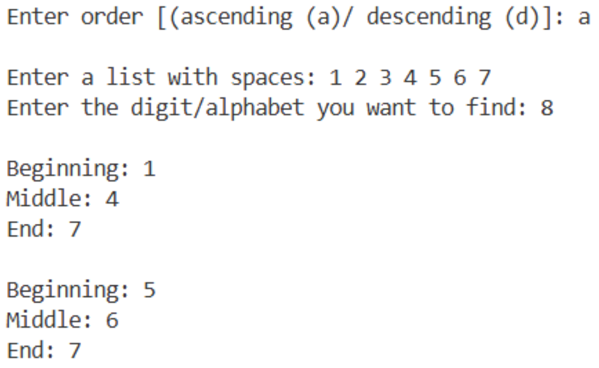


```
def binary_search(data, item):

    beg = 0
    end = len(data) - 1
    mid = int((beg + end) / 2)

    for i in range(len(data)):
        while beg <= end and data[mid] != item:
            print("\nBeginning:", data[beg])
            print("Middle:", data[mid])
            print("End:", data[end])
            if item < data[mid]:
                end = mid - 1
            else:
                beg = mid + 1

            mid = int((beg + end) / 2)

    if data[mid] == item:
        loc = mid
        return f"Item found at location '{loc}'"
```

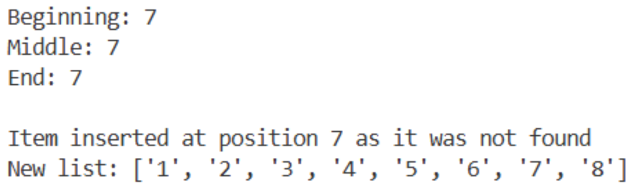


```
    else:
        data.append(item)
        sorted(data)
        return f"\nItem inserted at position {data.index(item)} " \
            f"as it was not found \nNew list: {data}"

order = input("Enter order [(ascending (a)/ descending (d)]: ").lower()
data = input('\nEnter a list with spaces: ').split()

if all(isinstance(item, int) for item in data):
    data = [int(item) for item in data]

elif all(isinstance(item, float) for item in data):
    data = [float(item) for item in data]

if order == 'a':
    if data != sorted(data):
        print("Data not sorted")
        exit()

if order == 'd':
    if data != sorted(data, reverse=True):
        print("Data not sorted")
        exit()

item = input("Enter the digit/alphabet you want to find: ")

search = binary_search(data, item)

print(search)
```

OUTPUT



```
Enter order [(ascending (a)/ descending (d)]: a

Enter a list with spaces: 1 2 3 4 5 6 7
Enter the digit/alphabet you want to find: 8

Beginning: 1
Middle: 4
End: 7

Beginning: 5
Middle: 6
End: 7
```



```
Beginning: 7
Middle: 7
End: 7

Item inserted at position 7 as it was not found
New list: ['1', '2', '3', '4', '5', '6', '7', '8']
```