
Experiment 3.1

Name :- Yash Gupta Uid :- 20BCS5009

Course :- CSE Section :- 709-A

Subject :- Programming in Python Subject Code :- 20CSP-259

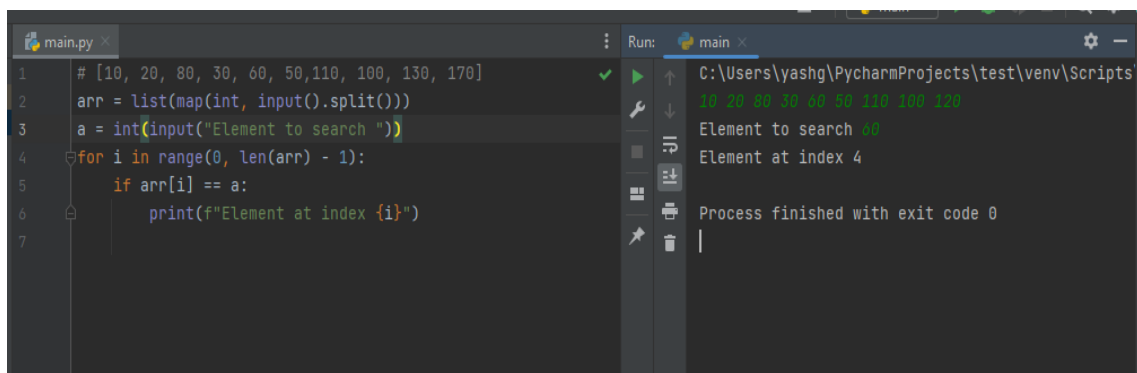
Aim:-

1. Python program to implement linear search.
2. Python program to implement bubble sort.
3. Python program to implement binary search without recursion.
4. Python program to implement selection sort.

Code:-

1.

```
def linearsearch(arr, x):  
    for i in range(len(arr)):  
        if arr[i] == x:  
            return i  
    return -1  
#[10, 20, 80, 30, 60, 50, 110, 100, 130, 170]  
arr=list(map(int,input().split()))  
x = input("Element to search")  
print(f"Element at index {linearsearch(arr,x)}")
```



```
main.py x  
1 # [10, 20, 80, 30, 60, 50, 110, 100, 130, 170]  
2 arr = list(map(int, input().split()))  
3 a = int(input("Element to search "))  
4 for i in range(0, len(arr) - 1):  
5     if arr[i] == a:  
6         print(f"Element at index {i}")  
7  
Run: main x  
C:\Users\yashg\PycharmProjects\test\venv\Scripts  
10 20 80 30 60 50 110 100 120  
Element to search 80  
Element at index 4  
Process finished with exit code 0
```

2.

```
def bubbleSort(lis):
```

```

length = len(lis)
for i in range(length):
    for j in range(length - i):
        a = lis[j]
        if a != lis[-1]:
            b = lis[j + 1]
            if a > b:
                lis[j] = b
                lis[j + 1] = a
    return lis
lis=list(map(int,input().split()))
a=bubbleSort(lis)
print(f"Sorted list {a}.")

```

```

1  def bubbleSort(lis):
2      length = len(lis)
3      for i in range(length):
4          for j in range(length - i):
5              a = lis[j]
6              if a != lis[-1]:
7                  b = lis[j + 1]
8                  if a > b:
9                      lis[j] = b
10                     lis[j + 1] = a
11             return lis
12
13
14  lis = list(map(int, input().split()))
15  a = bubbleSort(lis)
16  print(f"Sorted list {a}.")
17

```

Run: main x
C:\Users\yashg\PycharmProjects\test\venv\S
10 85 36 49 21 99 34
Sorted list [10, 21, 34, 36, 49, 85, 99].
Process finished with exit code 0

3.

```

def binary_search(my_list, elem):
    low = 0
    high = len(my_list) - 1
    mid = 0
    while low <= high:
        mid = (high + low) // 2
        if my_list[mid] < elem:
            low = mid + 1
        elif my_list[mid] > elem:
            high = mid - 1
        else:
            return mid
    return -1

```

```
my_list = [ 1, 9, 11, 21, 34, 54, 67, 90 ]
```

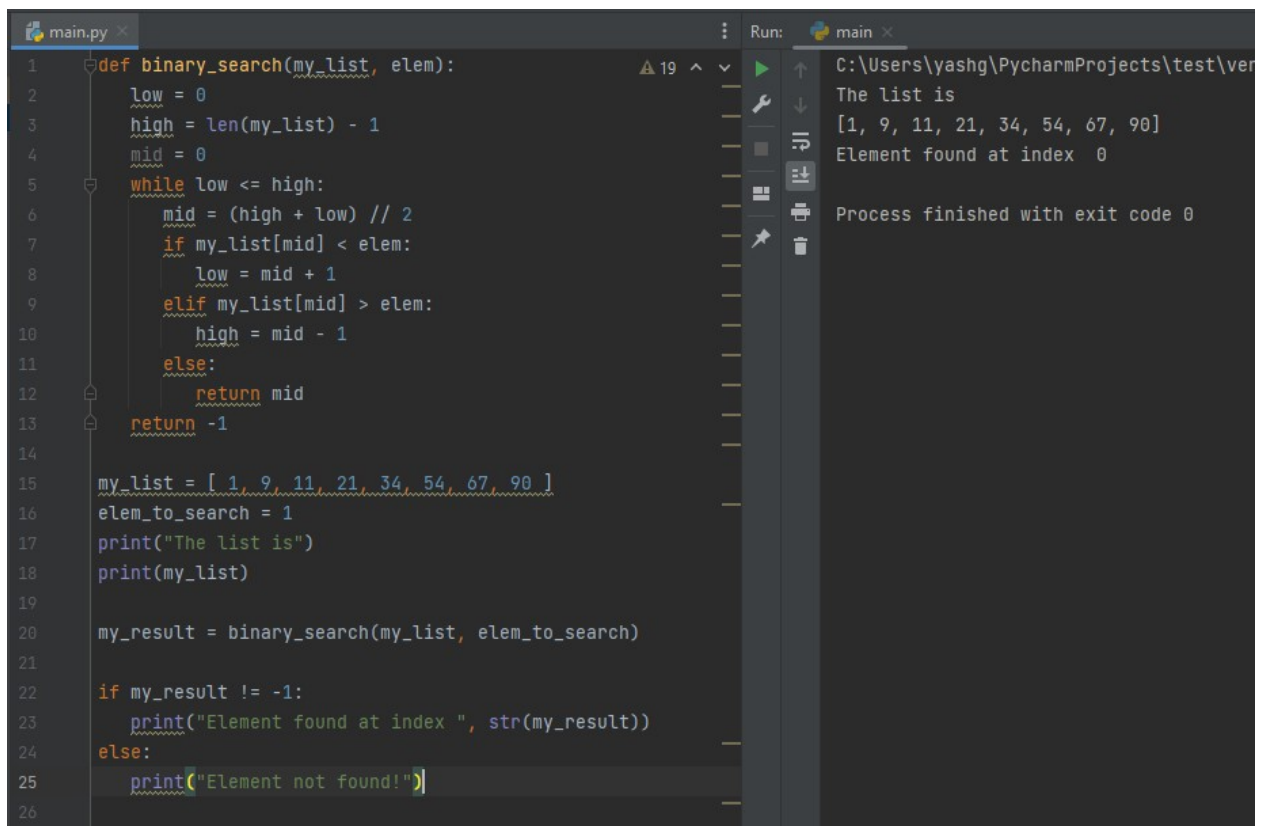
```

elem_to_search = 1
print("The list is")
print(my_list)

my_result = binary_search(my_list, elem_to_search)

if my_result != -1:
    print("Element found at index ", str(my_result))
else:
    print("Element not found!")

```



The screenshot shows a PyCharm IDE with a file named 'main.py'. The code defines a 'binary_search' function that takes a list and an element to search for. It uses a while loop to find the element's index. Below the function, a list 'my_list' is defined with values [1, 9, 11, 21, 34, 54, 67, 90], and 'elem_to_search' is set to 1. The function is called, and the result is printed. The output console on the right shows the execution results: 'The list is', '[1, 9, 11, 21, 34, 54, 67, 90]', and 'Element found at index 0'. The process finished with exit code 0.

```

1 def binary_search(my_list, elem):
2     low = 0
3     high = len(my_list) - 1
4     mid = 0
5     while low <= high:
6         mid = (high + low) // 2
7         if my_list[mid] < elem:
8             low = mid + 1
9         elif my_list[mid] > elem:
10            high = mid - 1
11        else:
12            return mid
13    return -1
14
15 my_list = [ 1, 9, 11, 21, 34, 54, 67, 90 ]
16 elem_to_search = 1
17 print("The list is")
18 print(my_list)
19
20 my_result = binary_search(my_list, elem_to_search)
21
22 if my_result != -1:
23     print("Element found at index ", str(my_result))
24 else:
25     print("Element not found!")
26

```

Run: main ×

C:\Users\yashg\PycharmProjects\test\ver
The list is
[1, 9, 11, 21, 34, 54, 67, 90]
Element found at index 0

Process finished with exit code 0

4.

```

A = [64, 25, 12, 22, 11]

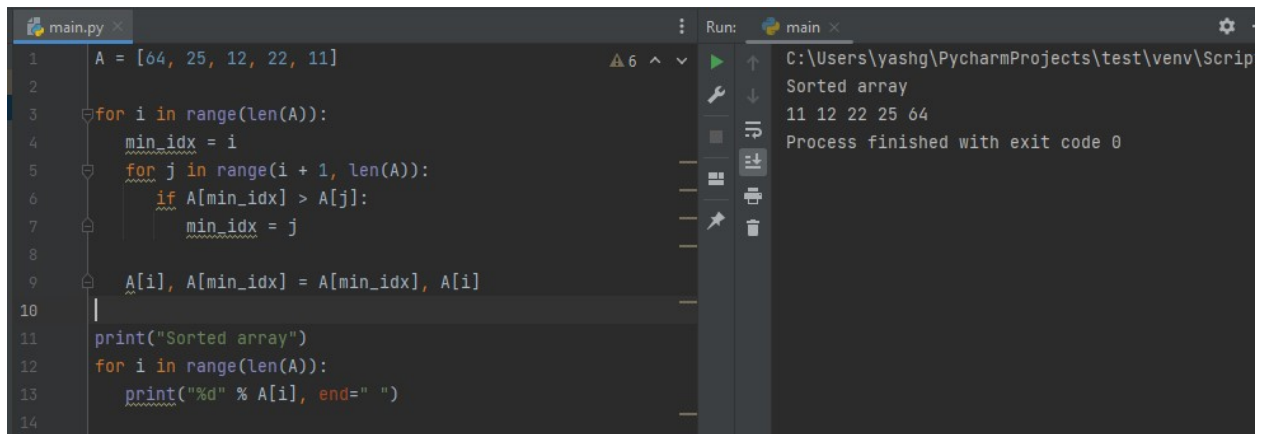
for i in range(len(A)):
    min_idx = i
    for j in range(i + 1, len(A)):
        if A[min_idx] > A[j]:
            min_idx = j

    A[i], A[min_idx] = A[min_idx], A[i]

print("Sorted array")

```

```
for i in range(len(A)):
    print("%d" % A[i], end=" ")
```



The screenshot shows a PyCharm IDE with a Python script in the editor and its execution output in the Run console. The script implements a selection sort algorithm on the array [64, 25, 12, 22, 11]. The output shows the sorted array [11, 12, 22, 25, 64].

```
main.py x
1  A = [64, 25, 12, 22, 11]
2
3  for i in range(len(A)):
4      min_idx = i
5      for j in range(i + 1, len(A)):
6          if A[min_idx] > A[j]:
7              min_idx = j
8
9      A[i], A[min_idx] = A[min_idx], A[i]
10
11 print("Sorted array")
12 for i in range(len(A)):
13     print("%d" % A[i], end=" ")
14
```

Run: main x
C:\Users\yashg\PycharmProjects\test\venv\Script
Sorted array
11 12 22 25 64
Process finished with exit code 0