1) Aim: Write a program to implement binary search method using recursive and non-recursive method.

   Program:

   a) With recursion

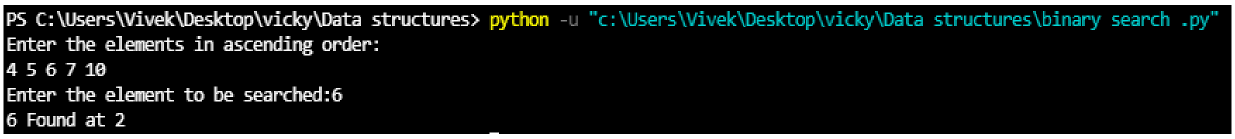
```
def binary_search(arr,l,h,key):
    if(h>=1):
        m = (l+h)//2
        if(arr[m]==key):
            return m
        elif(arr[m]<key):
            return binary_search(arr,m+1,h,key)
        else:
            return binary_search(arr,l,m-1,key)
print("Enter the elements in ascending order:")
l=[int(x) for x in input().split()]
key = int(input("Enter the element to be searched:"))
a=binary_search(l,0,len(l)-1,key)
print(str(key)+" Found at "+str(a))
```

   b) Without recursion

```
def binary_search(arr,l,h,key):
    while(l<=h):
        m = (l+h)//2
        if(arr[m]==key):
            return m
        elif(arr[m]<key):
            l = m+1
        else:
            h = m-1
print("Enter the elements in ascending order:")
l=[int(x) for x in input().split(' ')]
key = int(input("Enter the element to be searched:"))
a=binary_search(l,0,len(l)-1,key)
print(str(key)+" Found at "+str(a))
```

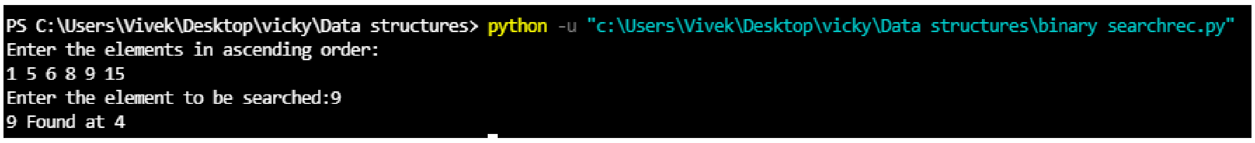
Output:

a) Without recursion:



```
PS C:\Users\Vivek\Desktop\vicky\Data structures> python -u "c:\Users\Vivek\Desktop\vicky\Data structures\binary search .py"
Enter the elements in ascending order:
4 5 6 7 10
Enter the element to be searched:6
6 Found at 2
```

b) With recursion:



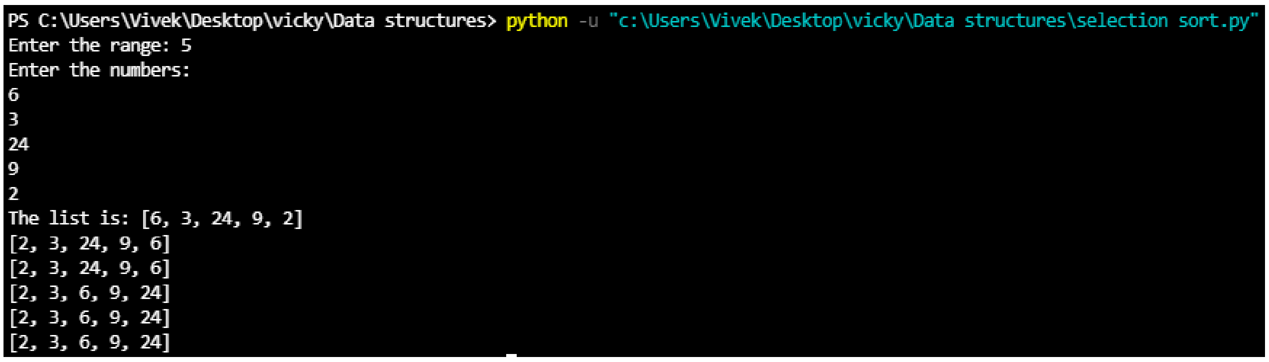
```
PS C:\Users\Vivek\Desktop\vicky\Data structures> python -u "c:\Users\Vivek\Desktop\vicky\Data structures\binary searchrec.py"
Enter the elements in ascending order:
1 5 6 8 9 15
Enter the element to be searched:9
9 Found at 4
```

2) **Aim**: Write a program to implement selection sort using recursive method.

Program:

```
def minindex(arr,i,j):
    if i==j:
        return i
    k = minindex(arr,i+1,j)
    return(i if arr[i]<arr[k] else k)
def selection_sort(arr,n,index=0):
    if index == n:
        return -1
    k = minindex(arr,index,n-1)
    if k!=index:
        arr[k],arr[index] = arr[index],arr[k]
    selection_sort(arr,n,index+1)
l = []
n = int (input("Enter the range: "))
print("Enter the numbers: ")
for i in range(n):
    x = int(input())
    l.append(x)
selection_sort(l,len(l))
print(l)
```

Output:



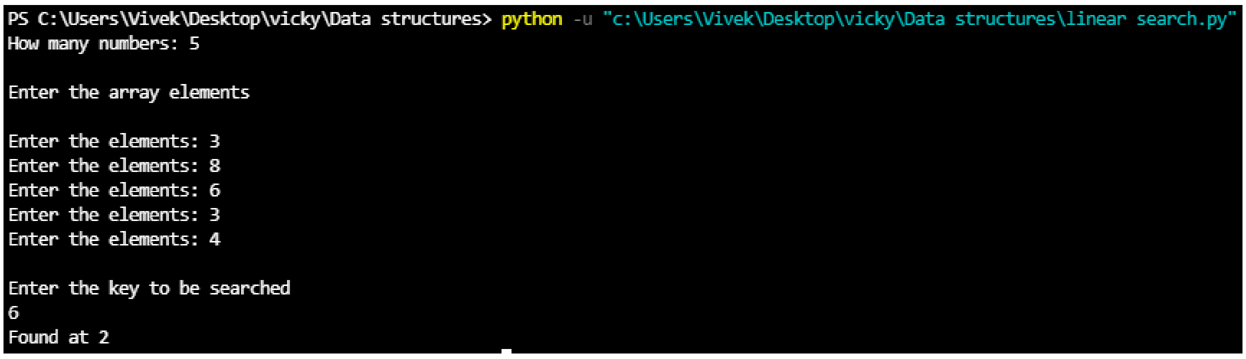
```
PS C:\Users\Vivek\Desktop\vicky\Data structures> python -u "c:\Users\Vivek\Desktop\vicky\Data structures\selection sort.py"
Enter the range: 5
Enter the numbers:
6
3
24
9
2
The list is: [6, 3, 24, 9, 2]
[2, 3, 24, 9, 6]
[2, 3, 24, 9, 6]
[2, 3, 6, 9, 24]
[2, 3, 6, 9, 24]
[2, 3, 6, 9, 24]
```

3) **Aim**: Write a program to implement linear search method using recursive method.

Program:

```
a=list()
FOUND=0
n=int(input("How many numbers: "))
if(n>10):
   print("\nToo many Numbers\n")
print("\nEnter the array elements\n")
for i in range(n):
   y=int(input("Enter the elements: "))
   a.append(y)
key=int(input("\nEnter the key to be searched\n"))
for i in range(n):
   if(a[i]==key):
      print("Found at",i)
      FOUND=1
      break
if(FOUND==0):
   print("\nNOT FOUND...")
```

Output:



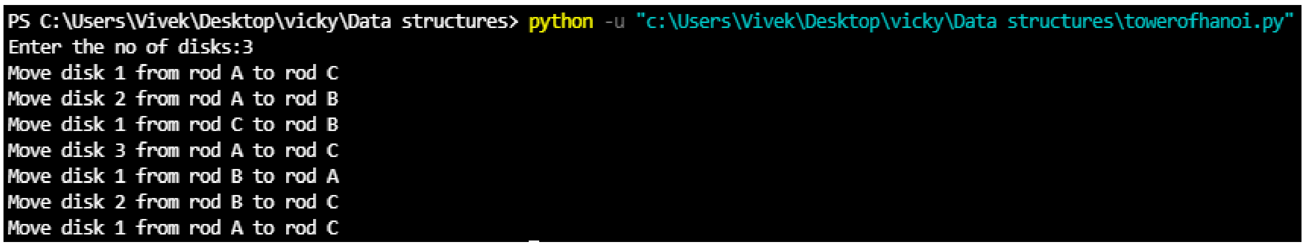
```
PS C:\Users\Vivek\Desktop\vicky\Data structures> python -u "c:\Users\Vivek\Desktop\vicky\Data structures\linear search.py"
How many numbers: 5

Enter the array elements

Enter the elements: 3
Enter the elements: 8
Enter the elements: 6
Enter the elements: 3
Enter the elements: 4

Enter the key to be searched
6
Found at 2
```

4) **Aim**: Write a program to implement towers of Hanoi

Program:

```
def TowerOfHanoi(n , from_rod, to_rod, aux_rod):
    if n == 1:
        print("Move disk 1 from rod",from_rod,"to rod",to_rod)
        return
    TowerOfHanoi(n-1, from_rod, aux_rod, to_rod)
    print("Move disk",n,"from rod",from_rod,"to rod",to_rod)
    TowerOfHanoi(n-1, aux_rod, to_rod, from_rod)
n = int(input("Enter the no of disks:"))
TowerOfHanoi(n, 'A', 'C', 'B')
```

Output:



```
PS C:\Users\Vivek\Desktop\vicky\Data structures> python -u "c:\Users\Vivek\Desktop\vicky\Data structures\towerofhanoi.py"
Enter the no of disks:3
Move disk 1 from rod A to rod C
Move disk 2 from rod A to rod B
Move disk 1 from rod C to rod B
Move disk 3 from rod A to rod C
Move disk 1 from rod B to rod A
Move disk 2 from rod B to rod C
Move disk 1 from rod A to rod C
```