

DATA STRUCTURE ASSIGNMENT 3.

ARUNTHATHI N

B.TECH AI &DS]

23102009.

1. Write a program to perform binary Search.

AIM:

To perform binary Search.

ALGORITHM:

Step1: Start

Step2: Set low to 0

Step3: Set high to the index of the last element in the array.

Step4: low is less than or equal to high.

Step5: End

PROGRAM:

```
def binary_search(start,end,a,k):
```

```
    for i in a:
```

```
        mid=(start+end)//2
```

```
        if a[mid]==k:
```

```
            print(mid)
```

```
            break
```

```
        elif a[mid]>k:
```

```

        return binary_search(start,mid,a,k)
    break
else:
    return binary_search(mid+1,end,a,k)
n=int(input('enter a number of element:'))
a=[]
for i in range(n):
    a.append(int(input('enter a items:')))
for i in range(0,n):
    for j in range(0,n-i-1):
        if a[j]>a[j+1]:
            a[j],a[j+1]=a[j+1],a[j]
print('the sorted list is :',a)

k=int(input('enter a search item:'))

print(binary_search(0,n-1,a,k))

```

OUTPUT:

```
IDLE Shell 3.12.2
File Edit Shell Debug Options Window Help

Python 3.12.2 (tags/v3.12.2:6abddd9, Feb 6 2024, 21:26:36) [MSC v.1937 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.

>>>
= RESTART: C:/Users/ASUS/OneDrive/Desktop/programming languages/data structures/binary search ARUNTHATHL.py
enter a number of element:7
enter a items:1
enter a items:2
enter a items:3
enter a items:4
enter a items:5
enter a items:6
enter a items:7
the sorted list is : [1, 2, 3, 4, 5, 6, 7]
enter a search item:7
6
None
>>>
```

RESULT:

To implement the binary search is completed successfully.

2. Write a program to perform Linear Search.

AIM:

To perform Linear Search.

ALGORITHM:

1. Start: Begin the algorithm.
2. Initialize: Set the initial index to 0.
3. Input: Read the array and the target value.
4. Iterate: Loop through the array, checking each element.
5. Check: If the current element matches the target, return the index.
6. Increment: Move to the next element.
7. Not Found: If the loop completes without finding the target, return -1.
8. End: End the algorithm.

PROGRAM:

```
n=int(input('enter a num:'))  
a=[]  
for i in range (n):  
    a.append(int(input('enter a list item:')))  
search_item=int(input('enter a search item:'))
```

```
found=False
```

```
for i in range(len(a)):
```

```
    if (a[i]==search_item):
```

```
        found=True
```

```
        break
```

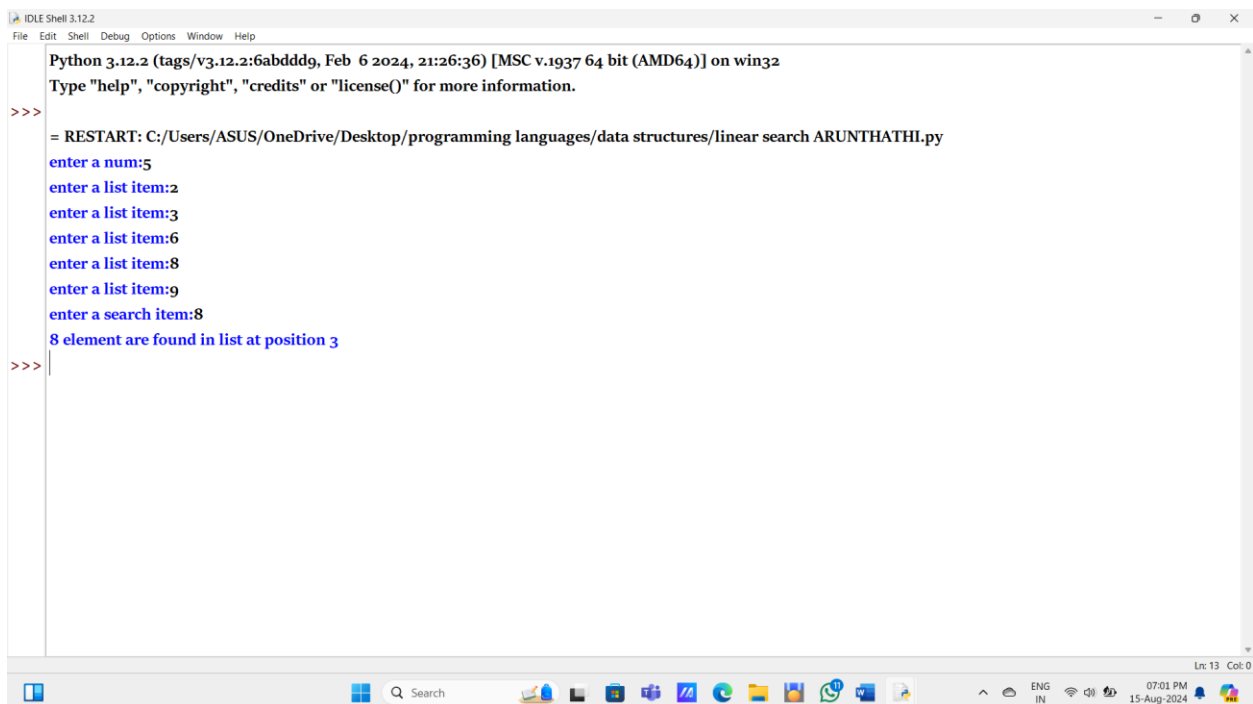
```
if found:
```

```
    print(search_item,'element are found in list at position',i)
```

```
else:
```

```
    print("search element is not found")
```

OUTPUT:



```
IDLE Shell 3.12.2
File Edit Shell Debug Options Window Help
Python 3.12.2 (tags/v3.12.2:6abddd9, Feb 6 2024, 21:26:36) [MSC v.1937 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
= RESTART: C:/Users/ASUS/OneDrive/Desktop/programming languages/data structures/linear search ARUNTHATHI.py
enter a num:5
enter a list item:2
enter a list item:3
enter a list item:6
enter a list item:8
enter a list item:9
enter a search item:8
8 element are found in list at position 3
>>>
```

RESULT:

Performed the linear search is Completed successfully.

3. Write a program to perform Selection Sort.

AIM:

To perform Selection Sort.

ALGORITHM:

1. Start: Begin the algorithm.
2. Initialize: Set the initial index i to 0.
3. Input: Read the array to be sorted.
4. Iterate: Loop through the array, assuming the current element is the minimum.
5. Inner Loop: Check each subsequent element to find the actual minimum.
6. Swap: Swap the current element with the found minimum element.
7. Increment: Move to the next element.

PROGRAM:

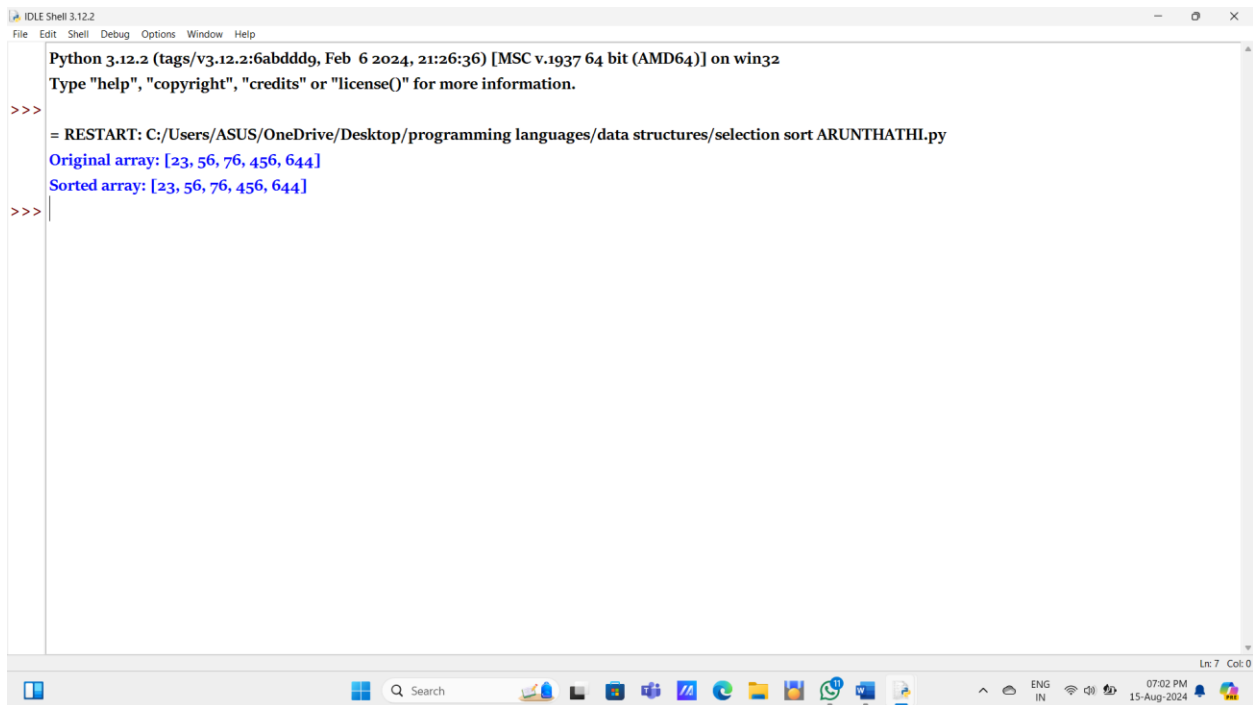
```
def selection_sort(arr):  
    n = len(arr)  
    for i in range(n):  
        min_index = i  
        for j in range(i + 1, n):  
            if arr[j] < arr[min_index]:
```

```
min_index = j
arr[i], arr[min_index] = arr[min_index], arr[i]
```

Example usage

```
arr = [45,49,9,87,65]
print("Original array:", arr)
selection_sort(arr)
print("Sorted array:", arr)
```

OUTPUT:

A screenshot of a Python IDLE Shell window. The window title is "IDLE Shell 3.12.2". The menu bar includes "File", "Edit", "Shell", "Debug", "Options", "Window", and "Help". The shell displays the following text: "Python 3.12.2 (tags/v3.12.2:6abdd9, Feb 6 2024, 21:26:36) [MSC v.1937 64 bit (AMD64)] on win32", "Type 'help', 'copyright', 'credits' or 'license()' for more information.", followed by a prompt ">>>". The user has entered a multi-line command: "= RESTART: C:/Users/ASUS/OneDrive/Desktop/programming languages/data structures/selection sort ARUNTHATHI.py". The shell has executed this command and displays the output: "Original array: [23, 56, 76, 456, 644]" and "Sorted array: [23, 56, 76, 456, 644]". The prompt ">>>" is shown again. The Windows taskbar is visible at the bottom, showing the Start button, Search bar, and various application icons. The system clock shows "07:02 PM 15-Aug-2024".

```
Python 3.12.2 (tags/v3.12.2:6abdd9, Feb 6 2024, 21:26:36) [MSC v.1937 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
= RESTART: C:/Users/ASUS/OneDrive/Desktop/programming languages/data structures/selection sort ARUNTHATHI.py
Original array: [23, 56, 76, 456, 644]
Sorted array: [23, 56, 76, 456, 644]
>>>
```

RESULT:

To implement the selection sort is completed successfully.

4. Write a program to perform Insertion Sort.

AIM:

To perform Insertion Sort.

ALGORITHM:

1. Start: Begin the algorithm.
2. Initialize: Set the initial index i to 1.
3. Input: Read the array to be sorted.
4. Iterate: Loop through the array, starting from the second element.
5. Set: Store the current element in a variable key .
6. Inner Loop: Compare and shift elements in the sorted portion of the array.
7. Insert: Place the key in its correct position.
8. End: End the algorithm when the entire array is sorted.

PROGRAM:

```
def insertion_sort(arr):  
    for i in range(1, len(arr)):  
        key = arr[i]  
        j = i - 1  
        while j >= 0 and arr[j] > key:  
            arr[j + 1] = arr[j]
```



```
j -= 1  
arr[j + 1] = key
```

Example usage

```
arr = [77,87,97,67,57]  
print("Original array:", arr)  
insertion_sort(arr)  
print("Sorted array:", arr)
```

OUTPUT:

```
IDLE Shell 3.12.2  
File Edit Shell Debug Options Window Help  
Python 3.12.2 (tags/v3.12.2:6abddd9, Feb 6 2024, 21:26:36) [MSC v.1937 64 bit (AMD64)] on win32  
Type "help", "copyright", "credits" or "license()" for more information.  
>>>  
= RESTART: C:/Users/ASUS/OneDrive/Desktop/programming languages/data structures/insertion sort ARUNTHATHI.py  
Original array: [12, 34, 32, 36, 35, 3]  
Sorted array: [3, 12, 32, 34, 35, 36]  
>>>
```

RESULT:

To perform Insertion Sort is completed successfully.