**“DATA STRUCTURE AND ALGORITHM LAB”**

**A LAB RECORD SUBMITTED IN PARTIAL FULFILLMENT**

**OF THE REQUIREMENTS FOR THE SUBJECT**

**“DATA STRUCTURE AND ALGORITHM”**

**OF**

**Bachelor of Technology (Computer Science)**

**Submitted by:**

**Md merajul haque**

**B.Tech. (Computer Science) 2nd Year**

**Roll Number: 22BLCS005HY**

**Enrollment Number: A191069**

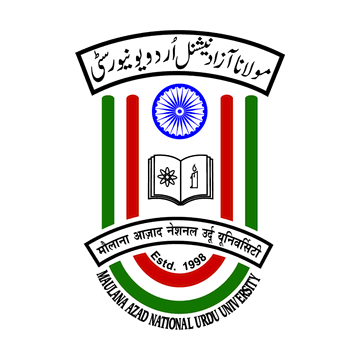
**Submitted to:**

**MOHTESHAM PASHA QUADRI**

**Assistant Professor**

**Department of Computer Science & Information Technology**

**Maulana Azad National Urdu University, Hyderabad**

****

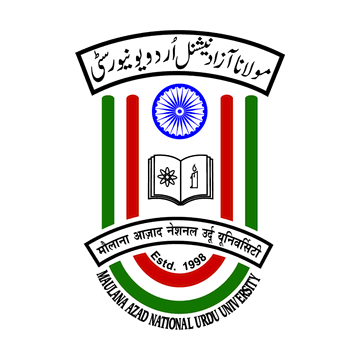
**Department of Computer Science & Information Technology**

**Maulana Azad National Urdu University, Hyderabad**

**Maulana Azad National Urdu University**

**Gachibowli, Hyderabad, Telangana-500032 (India)**

**(Accredited with “A+” Grade by NAAC )**

****

**Certificate**

This is to certify that the lab record file by **MD MERAJUL HAQUE** bearing Enrollment Number **A191069** submitted in partial fulfillment of the requirements for the subject **“DATA STRUCTURE AND ALGORITHM LAB”** with course code **“BTCS360PCP”** in **Bachelor of Technology** (Computer Science) **3rd Semester** during 2022-23 at the **Department of Computer Science & Information Technology** is a bonafide laboratory work carried out by him under my supervision.

The results presented in this file have been verified and are found to be satisfactory.

**Signature of Internal Examiner Signature of External Examiner**

**INDEX / انڈیکس**

**LAB PRACTICAL RECORD / لیب عملی ریکار**

|  |  |  |
| --- | --- | --- |
| **S.No.** | **NAME OF EXPERIMRNT** | **PAGE NUMBER** |
| **1** | **Implement the Array Operation ( Insertion )** | **04 - 08** |
| **2** | **Implement the Array Operation ( Deletion )** |  |
| **3** | **Implementation of Stack Operation ( Push, Pop and Peak )** |  |
| **4** | **Implementation of Infix to Postfix** |  |
| **5** | **Implementation of Simple Queue Operation ( Enqueue and Dequeue )** |  |
| **6** | **Implementation of Circular Queue Operation ( Enqueue and Dequeue )** |  |
| **7** | **Sorting Technique ( Bubble Sort )** |  |
| **8** | **Sorting Technique ( Selection Sort )** |  |
| **9** | **Sorting Technique ( Insertion Sort )** |  |
| **10** | **Sorting Technique ( Quick Sort )** |  |

**) :- 01*پروگرام* No ( Program**

#### Aim (*مقصد*) :- IMPLEMENT THE ARRAY OPERATION (INSERTION)

**) :-*الگورتھم/طریقہ کار*Algorithm (**

### INSERTION AT THE END OF ARRAY:

Step 1: if UB=MAX, then array is overflow Step 2: Read DATA

Step 3: UB=UB+1

Step 4: arr[UB]=DATA

### INSERTION AT THE BEGINNING OF ARRAY:

Step 1: if UB=MAX,then array is overflow Step 2: Read DATA

Step 3: k=UB

Step 4: Repeat step 5 while k>=LB Step 5: arr[K+1]=arr[k]

K=k-1

Step 6: arr[LB]=DATA

Step 7: Stop

### INSERTION AT THE GIVEN POSITION/LOCATION OF ARRAY:

Step 1: if UB=MAX,then array is overflow Step 2: Read DATA and LOCATION

Step 3: k=UB

Step 4: Repeat step 5 while k>=LOCATION Step 5: arr[K+1]=arr[k]

K=k-1

Step 6: arr[LOCATION]=DATA

Step 7: Stop

**Code(*کوڈ*) :-**

**#include<stdio.h>**

**#include <stdlib.h>**

**#define max 100**

**void insertion();**

**void insertionEnd();**

**void insertionBegin();**

**void insertionLoc();**

**void display();**

**struct array{**

**int ub;**

**int lb;**

**int a[max];**

**int size;**

**}a;**

**void main(){**

**int choice;**

**a.lb = -1;**

**a.ub = -1;**

**printf("Enter the size of Array : ");**

**scanf("%d",&a.size);**

**insertion();**

**}**

**void insertion(){**

**int choice,d;**

**do{**

**printf("\n1. Insert at the End\n2. Insert at the Begin\n3. Insert at the Location\n4. Display\n5. Exit\n ");**

**scanf("%d",&choice);**

**switch (choice){**

**case 1:**

**insertionEnd();**

**display();**

**break;**

**case 2:**

**insertionBegin();**

**display();**

**break;**

**case 3:**

**insertionLoc();**

**display();**

**break;**

**case 4:**

**display();**

**break;**

**case 5:**

**exit(0);**

**default:**

**printf("Wrong choice!!!");**

**break;**

**}**

**}while(choice!=5);**

**}**

**void insertionEnd(){**

**int data;**

**if(a.ub >= a.size){**

**printf("Array is Overflow!!");**

**}else{**

**printf("Enter element to be inserted : ");**

**scanf("%d",&data);**

**a.ub++;**

**a.a[a.ub] = data;**

**}**

**}**

**void insertionBegin(){**

**int data,i;**

**if(a.ub >= a.size){**

**printf("Array is Overflow!!");**

**}else{**

**printf("Enter element to be inserted : ");**

**scanf("%d",&data);**

**a.ub++;**

**for(i=0;i<=a.ub;i++){**

**a.a[i+1] = a.a[i];**

**}**

**a.a[0] = data;**

**}**

**}**

**void insertionLoc(){**

**int data,loc,i;**

**if(a.ub >= a.size){**

**printf("Array is Overflow!!");**

**}else{**

**printf("Enter element to be inserted : ");**

**scanf("%d",&data);**

**printf("Enter position : ");**

**scanf("%d",&loc);**

**a.ub++;**

**for(i=loc;i<=a.ub;i++){**

**a.a[i+1] = a.a[i];**

**}**

**a.a[loc] = data;**

**}**

**}**

**void display(){**

**int i;**

**if(a.ub == -1){**

**printf("No Elements!!");**

**}else{**

**printf("Array Elements : ");**

**for(i=0;i<=a.ub;i++){**

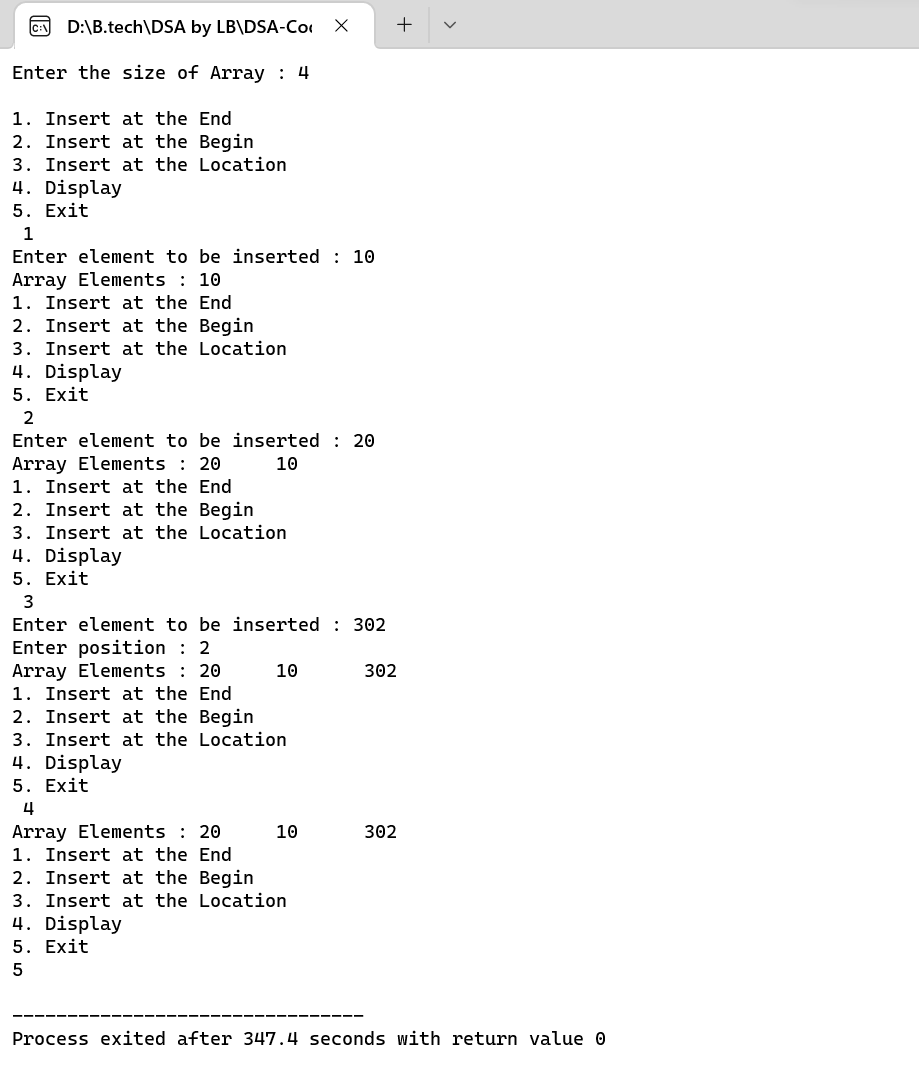
**printf("%d\t",a.a[i]);**

**}**

**}**

**}**

**Output(*نتیجہ*):-**

****

**) :- 02*پروگرام* No ( Program**

#### Aim (*مقصد*) :- **IMPLEMENT THE ARRAY OPERATION ( DELETION )**

**) :-*الگورتھم/طریقہ کار*Algorithm (**

**BEGINNING:**

Step 1: if UB==0, the array is underflow Step 2: k=LB

Step 3: Repeat the step 4 while k<UB Step 4: arr[k]=arr[k+1]

K=k+1

Step 5: arr[UB]=NULL

UB+UB-1

Step 6: stop

### DELETION AT END:

Step 1: if UB==0 then array is underflow Step 2: arr[UB]=NULL

UB=UB-1

Step 3: stop

### DELETION AT THE GIVEN LOCATION:

Step 1: if UB==0, the array is underflow Step 2: Read LOCATION

K=LOCATION

Step 3: Repeat the step 4 while k<UB Step 4: arr[k]=arr[k+1]

K=k+1

Step 5: arr[UB]=NULL

UB+UB-1

Step 6: stop

**Code(*کوڈ*) :-**

**#include<stdio.h>**

**#include <stdlib.h>**

**#define max 100**

**void deletion();**

**void deletionEnd();**

**void deletionBegin();**

**void deletionLoc();**

**void display();**

**struct array{**

**int ub;**

**int lb;**

**int a[max];**

**int size;**

**}a;**

**void main(){**

**int choice,i;**

**a.lb = -1;**

**printf("Enter size of Array : ");**

**scanf("%d",&a.size);**

**printf("Enter Elements : \n");**

**for(i=0;i<a.size;i++){**

**scanf("%d",&a.a[i]);**

**}**

**a.ub = a.size-1;**

**display();**

**deletion();**

**}**

**void deletion(){**

**int choice,d;**

**do{**

**printf("\n1. Delete at the End\n2. Delete at the Begin\n3. Delete at the Location\n4. Display\n5. Exit\n ");**

**scanf("%d",&choice);**

**switch (choice){**

**case 1:**

**deletionEnd();**

**display();**

**break;**

**case 2:**

**deletionBegin();**

**display();**

**break;**

**case 3:**

**deletionLoc();**

**display();**

**break;**

**case 4:**

**display();**

**break;**

**case 5:**

**exit(0);**

**default:**

**printf("Wrong choice!!!");**

**break;**

**}**

**}while(choice!=5);**

**}**

**void deletionEnd(){**

**int data;**

**if(a.ub == a.lb){**

**printf("Array is Underflow!!");**

**}else{**

**printf("Deleted Element : %d\n",a.a[a.ub]);**

**a.ub--;**

**}**

**}**

**void deletionBegin(){**

**int data,i;**

**if(a.ub == a.lb){**

**printf("Array is Underflow!!");**

**}else{**

**printf("Deleted Element : %d\n",a.a[0]);**

**for(i=0;i<a.ub;i++){**

**a.a[i] = a.a[i+1];**

**}**

**a.ub--;**

**}**

**}**

**void deletionLoc(){**

**int data,loc,i;**

**if(a.ub == a.lb){**

**printf("Array is Underflow!!");**

**}else{**

**printf("Enter position : ");**

**scanf("%d",&loc);**

**printf("Deleted Element : %d\n",a.a[loc-1]);**

**for(i=loc-1;i<a.ub;i++){**

**a.a[i] = a.a[i+1];**

**}**

**a.ub--;**

**}**

**}**

**void display(){**

**int i;**

**if(a.ub == -1){**

**printf("No Elements!!");**

**}else{**

**printf("Array Elements : ");**

**for(i=0;i<=a.ub;i++){**

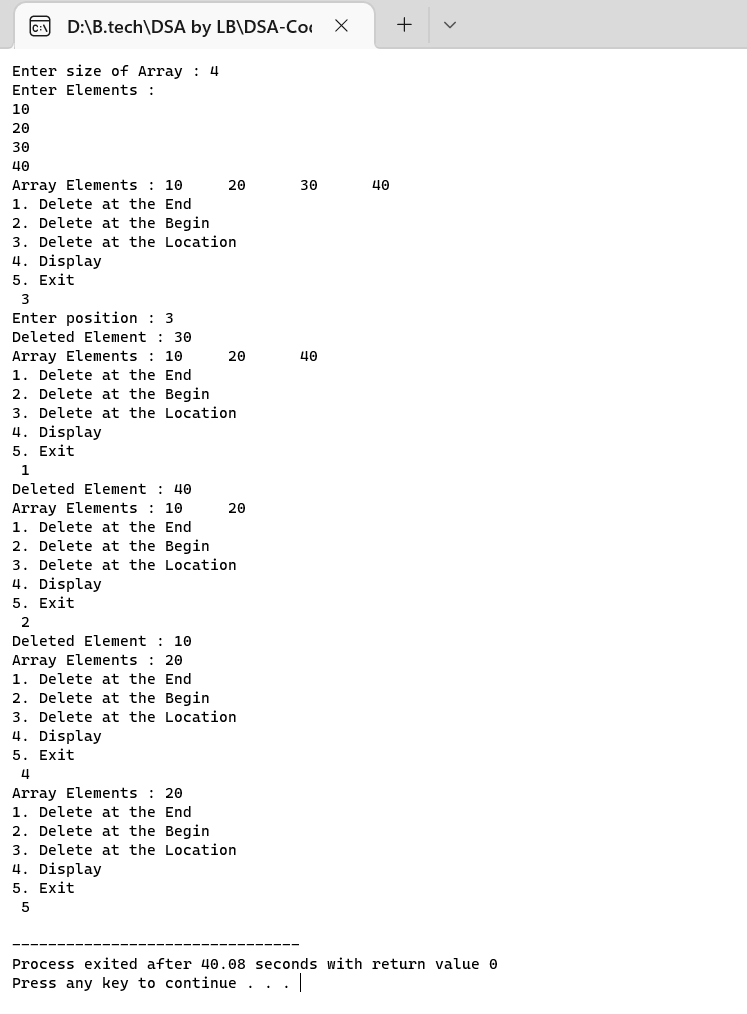
**printf("%d\t",a.a[i]);**

**}**

**}**

**}**

**Output(*نتیجہ*):-**

****

**) :- 03*پروگرام* No ( Program**

#### Aim (*مقصد*) :- **Implementation of Stack Operation ( Push, Pop and Peak )**

**) :-*الگورتھم/طریقہ کار*Algorithm (**

**Push Operation**

The process of putting a new data element onto stack is known as a Push Operation. Push operation involves a series of steps −

**Step 1** − Checks if the stack is full.

**Step 2** − If the stack is full, produces an error and exit.

**Step 3** − If the stack is not full, increments **top** to point next empty space. **Step 4** − Adds data element to the stack location, where top is pointing. **Step 5** − Returns success.

**POP OPERATION**

**Step 1** − Checks if the stack is empty.

**Step 2** − If the stack is empty, produces an error and exit.

**Step 3** − If the stack is not empty, accesses the data element at which **top** is pointing.

**Step 4** − Decreases the value of top by 1.

**Step 5** − Returns success.

### PEAK OPERATION

Step 1**:**begin procedure peek Step 2: return stack[top] Step 3: end procedure

**Code(*کوڈ*) :-**

**#include<stdio.h>**

**#define max 100**

**void push();**

**void pop();**

**void display();**

**void peek();**

**struct stack{**

**int top;**

**int arr[max];**

**}s;**

**void main(){**

**int x;**

**s.top = -1;**

**do{**

**printf("\n1. Push\n2. Pop\n3. Display\n4. Peek\n5. Exit\n");**

**printf("Choose Option : ");**

**scanf("%d",&x);**

**switch(x){**

**case 1:**

**push();**

**display();**

**break;**

**case 2:**

**pop();**

**display();**

**break;**

**case 3:**

**display();**

**break;**

**case 4:**

**peek();**

**break;**

**case 5:**

**break;**

**}**

**}while(x!=5);**

**}**

**void push(){**

**int data;**

**if(s.top == max-1){**

**printf("Stack Overflow!!");**

**}**

**else{**

**printf("Enter the element to be pushed : ");**

**scanf("%d",&data);**

**s.top++;**

**s.arr[s.top] = data;**

**}**

**}**

**void pop(){**

**int data;**

**if(s.top == -1){**

**printf("Stack Underflow!!\n");**

**}**

**else{**

**data = s.arr[s.top];**

**s.top--;**

**printf("Popped Element is %d\n",data);**

**}**

**}**

**void display(){**

**int i=0;**

**if(s.top == -1){**

**printf("Stack Underflow!!");**

**}else{**

**printf("Elements are : ");**

**for(i=0;i<=s.top;i++){**

**printf("%d\t",s.arr[i]);**

**}**

**}**

**}**

**void peek(){**

**if(s.top == -1){**

**printf("Stack Underflow!!");**

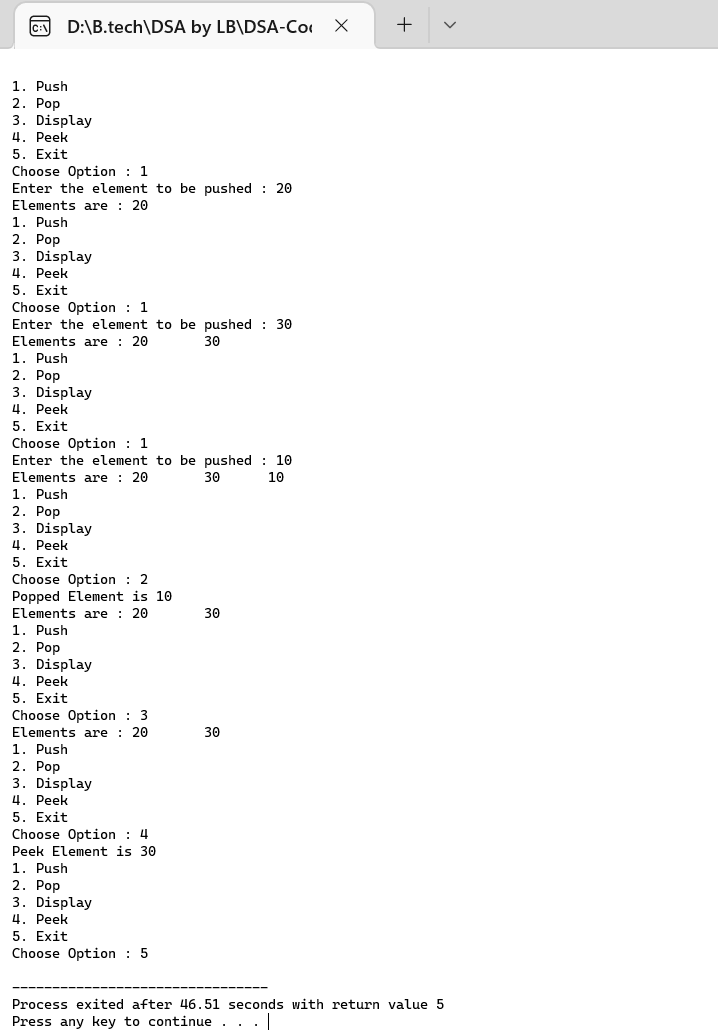
**}else{**

**printf("Peek Element is %d",s.arr[s.top]);**

**}**

**}**

**Output(*نتیجہ*):-**

****

**) :- 0*پروگرام* No ( Program**

#### **Aim (*مقصد*) :-**

**C program for implementation of Bubble sort**

**) :-*الگورتھم/طریقہ کار*Algorithm (**

1. **begin BubbleSort(arr)**
2. **for all array elements**
3. **if arr[i] > arr[i+1]**
4. **swap(arr[i], arr[i+1])**
5. **end if**
6. **end for**
7. **return arr**
8. **end BubbleSort**

**Code(*کوڈ*) :-**

**// C program for implementation of Bubble sort**

**#include<stdio.h>**

**void swap(int \*xp, int \*yp)**

**{**

**int temp = \*xp;**

**\*xp = \*yp;**

**\*yp = temp;**

**}**

**void bubbleSort(int arr[], int n)**

**{**

**int i, j;**

**for (i = 0; i<n-1; i++)**

**for (j = 0; j<n-i-1; j++)**

**if (arr[j] >arr[j+1])**

**swap(&arr[j], &arr[j+1]);**

**}**

**void printArray(int arr[], int size)**

**{**

**int i;**

**for (i=0; i<size; i++)**

**printf("%d ", arr[i]);**

**printf("\n");**

**}**

**int main()**

**{**

**int arr[100], n,i;**

**printf("Enter the no. of element you want to Sort. \n");**

**scanf("%d", &n);**

**printf("Now Enter the Element : \n");**

**for(i = 0; i<n; i++){**

**scanf("%d", &arr[i]);**

**}**

**bubbleSort(arr, n);**

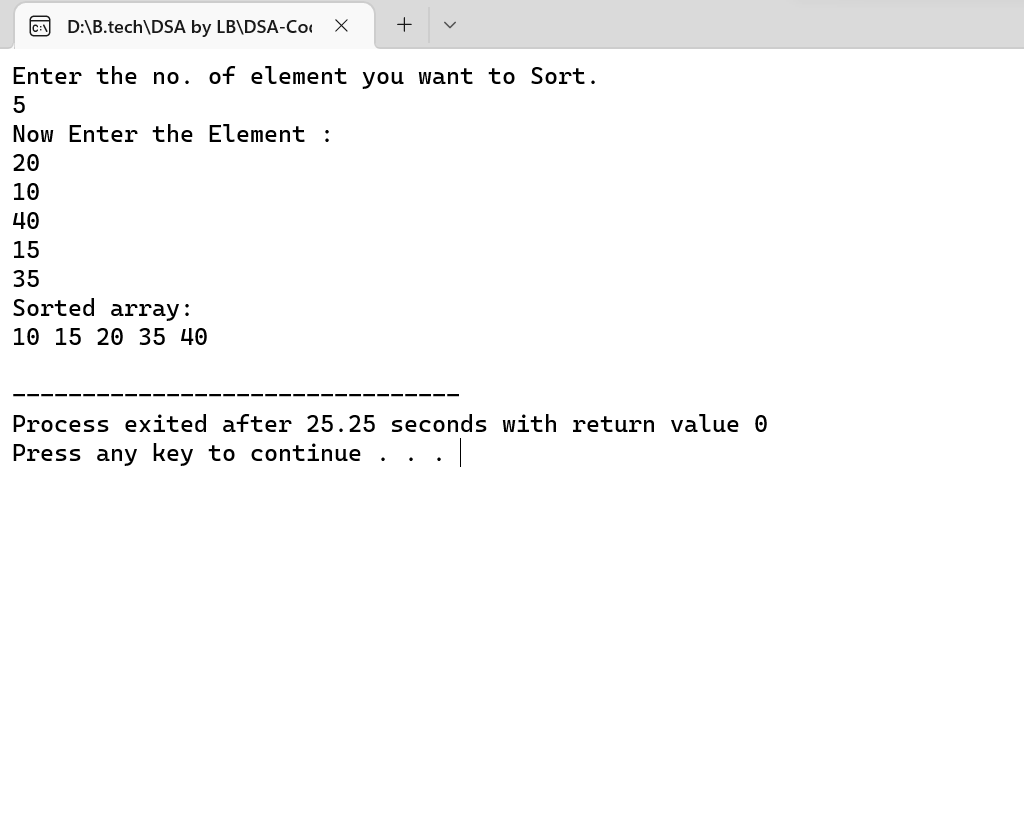
**printf("Sorted array: \n");**

**printArray(arr, n);**

**return 0;**

**}**

**Output(*نتیجہ*):-**

****

**) :- 01*پروگرام* No ( Program**

#### Aim (*مقصد*) :-

**) :-*الگورتھم/طریقہ کار*Algorithm (**

**Output(*نتیجہ*):-**