



NATIONAL INSTITUTE OF TECHNOLOGY PUDUCHERRY

(An Institution of National Importance under MHRD, Govt. of India)

KARAikal – 609 609

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

Roll Number: CS19B1009

Name : ARUN KUMAR R

Semester: II

Class: B TECH

Subject Code: CS106

Subject Name: DATA STRUCTURES LABORATORY

1. BUBBLE SORT

Date:11.12.19

AIM:

To find the c program to sort the given array using bubble sort algorithm

ALGORITHM:

1. Start the program.
2. Declare the variables.
3. Get the input from the user
4. Compare the adjacent elements from beginning and swap the element if the element is bigger than adjacent element.
5. Repeat the above step using loop for n -1 steps.
6. Output the result.
7. End the program.

PROGRAM:

```
#include<stdio.h>
int Bubble_sort(int x[], int n);
int main()
{
    int x[100], n, i;

    printf("Enter the number of elements\n");
    scanf("%d", &n );

    printf("Enter the elements of the array\n");
    for (i = 0 ; i < n; i++)
    {
        scanf("%d",&x[i] );
    }

    Bubble_sort(x,n);

    printf("The sorted array is \n");
    for(i = 0; i < n; i++)
    {
        printf("%d\n", x[i]);
    }

    return 0;
}

int Bubble_sort(int x[], int n)
{
    int i, j, temp;
```

```
for(i = 0 ; i < n - 1 ; i++)  
{  
    for (j = 0; j < n - 1 - i ; j++)  
    {  
        if( x[j] > x[j + 1])  
        {  
            temp = x[j];  
            x[j] = x[j + 1];  
            x[j + 1] = temp;  
        }  
    }  
}
```

OUTPUT:

```
Enter the number of elements  
5  
Enter the elements of the array  
3 4 2 6 1  
The sorted array is  
1  
2  
3  
4  
6
```

RESULT:

The program was executed successfully.

2.SELECTION SORT

DATE:11.12.19

AIM:

To find the c program to sort a given array using selection sort algorithm.

ALGORITHM:

1. Start the program.
2. Declare the variables.
3. Get the input from the user.
4. Using loop and if statement get the index of largest element and swap it to last.
5. Repeat the above step for n -1 iteration.
6. Output the result.
7. End the program.

PROGRAM:

```
#include <stdio.h>

void selection_sort(int x[], int n);

void main()
{

    int x[100], n, i;

    printf("Enter the number of elements\n");
```

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

```
printf("Enter the elements of the array\n");
```

```
for (i = 0 ; i < n; i++)
```

```
{
```

```
    scanf("%d",&x[i] );
```

```
}
```

```
selection_sort(x,n);
```

```
printf("The sorted array is \n");
```

```
for(i = 0; i < n; i++)
```

```
{
```

```
    printf("%d\n", x[i]);
```

```
}
```

```
}
```

```
void selection_sort(int x[], int n)
```

```
{
```

```
    int i, j, max, temp;
```

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

```
    {
```

```
        max = 0;
```

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

```
        {
```

```
            if( x[j] > x[max])
```

```
            {
```

```
                max = j;
```

```
            }
```

```
        }
```

```
        temp = x[max];
```

```
        x[max] = x[n - 1 - i];  
        x[n - 1 - i] = temp;  
    }  
  
}
```

OUTPUT:

```
Enter the number of elements  
5  
Enter the elements of the array  
2 4 1 5 6  
The sorted array is  
1  
2  
4  
5  
6
```

RESULT:

The program was executed successfully.

3.INSERTION SORT

DATE:11.12.19

AIM:

To find the c program to sort the given array using insertion sort algorithm.

ALGORITHM:

1. Start the program.
2. Declare the variables.
3. Input the array from the user.
4. Select the second element as a key and insert it left side at correct position.
5. Repeat the above step for other elements using loop.
6. Output the result.
7. End the program.

PROGRAM:

```
#include<stdio.h>
int insertsort(int a[], int n)
{
    int key, i, j;
```

```

for( i =1 ; i < n; i++)
{
    key = a[i];
    j = i-1;
    while(j >= 0 && key < a[j] )
    {
        a[j+1] = a[j];
        j--;
    }
    a[j+1] = key;
}
}

```

```

void main()
{

```

```

    int x[100], n, i;

```

```

    printf("Enter the number of elements\n");
    scanf("%d", &n );

```

```

    printf("Enter the elements of the array\n");
    for (i = 0 ; i < n; i++)
    {
        scanf("%d",&x[i] );
    }

```

```

    insertsort(x,n);

```

```

    printf("The sorted array is \n");
    for(i = 0; i < n; i++)
    {
        printf("%d\n", x[i]);
    }

```


}

}

OUTPUT:

```
Enter the number of elements
5
Enter the elements of the array
3 1 4 5 2
The sorted array is
1
2
3
4
5
```

RESULT:

The program was executed successfully.

4.QUICK SORT

DATE:11.12.19

AIM:

To find the c program to sort the given array using quick sort algorithm.

ALGORITHM:

1. Start the program.
2. Declare the variables.
3. Input the variables.
4. Select a pivot element in the array, and partition the array into an array with element lesser than pivot swapping elements to left of pivot and greater than pivot swapping to right of pivot.
5. Repeat the above step for each partition using recursion.
6. Output the result.
7. End the program.

PROGRAM:

```
#include<stdio.h>
int partition(int a[], int l, int h)
{
    int pivot = a[h], i, j = l;
    i = l-1;
    while(j <= h-1)
    {
        if(pivot > a[j])
        {
            i++;
            int temp = a[i];
            a[i] = a[j];
```

```
        a[j] = temp;
    }
    j++;
}
```

```
int temp = a[i+1];
a[i+1] = a[h];
a[h] = temp;
```

```
    return i + 1;
}
int quicksort(int a[], int l, int h)
{
    int p;

    if(l < h)
    {
        p = partition(a,l,h);
        quicksort(a,l,p-1);
        quicksort(a, p+1, h);
    }
}
```

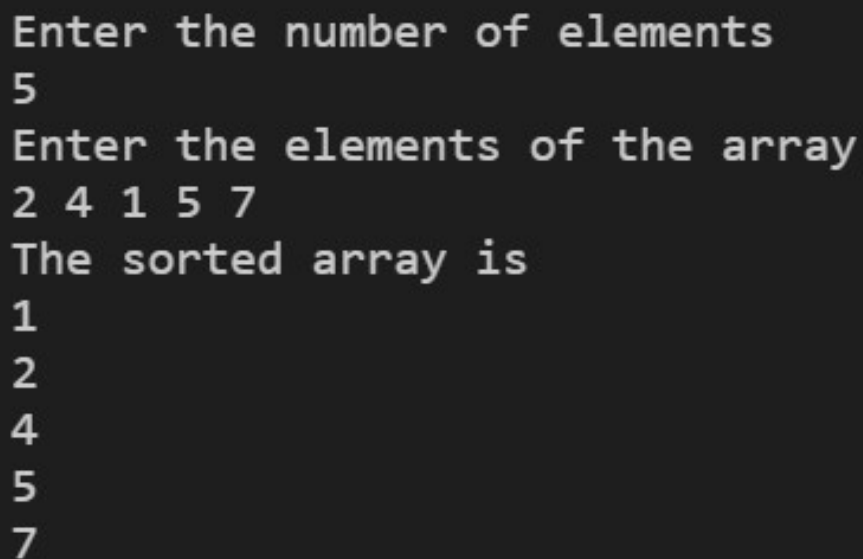
```
void main()
{
    int x[100], n, i;

    printf("Enter the number of elements\n");
    scanf("%d", &n );

    printf("Enter the elements of the array\n");
    for (i = 0 ; i < n; i++)
    {
```

```
    scanf("%d",&x[i] );  
}  
  
quicksort(x,0,n-1);  
  
printf("The sorted array is \n");  
for(i = 0; i < n; i++)  
{  
    printf("%d\n", x[i]);  
}  
  
}
```

OUTPUT:

A screenshot of a terminal window with a dark background and light gray text. The text shows the program's execution: it prompts for the number of elements (5), then for the elements of the array (2 4 1 5 7), and finally displays the sorted array (1 2 4 5 7).

```
Enter the number of elements  
5  
Enter the elements of the array  
2 4 1 5 7  
The sorted array is  
1  
2  
4  
5  
7
```

RESULT:

The program was executed successfully.