DATA STRUCTURE

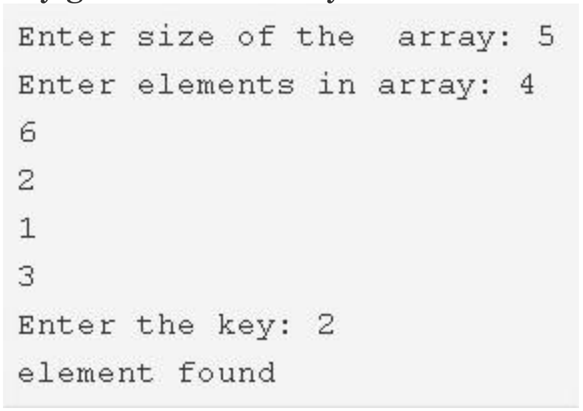
PRACTICAL 1

PRACTICAL 1A

Write a program to store the element using 1 d array and perform the operation like searching, sorting and traversing the element.

Searching

As we all know, an array is a collection or sequential arrangement of elements in any given order. Arrays form the basics of the C programming language.



```
Enter size of the  array: 5
Enter elements in array: 4
6
2
1
3
Enter the key: 2
element found
```

As you can see in the photo uploaded above, the elements are first entered in no particular order.
The elements entered here are 4, 6, 2, 1 and 3.
The target element is mentioned here as well. The target element here is 2.
Once the element is present in the array, it will say "element found".

Algorithm

1. Read the array size and store that value into the variable n.

2) Read the entered elements and store those elements into an array a[] using scanf statement and the for loop. scanf("%d",&a[i]) indicates scanf statement reads the entered element and assigned that element to a[i].
3) Read the key which we want to search in an array.
4) Compare the key with each element of the array as a[i]==key and print element found, if the key matches with any element of the array otherwise print element not found.
C Program To Search An Element In An Array Using Standard Method
C

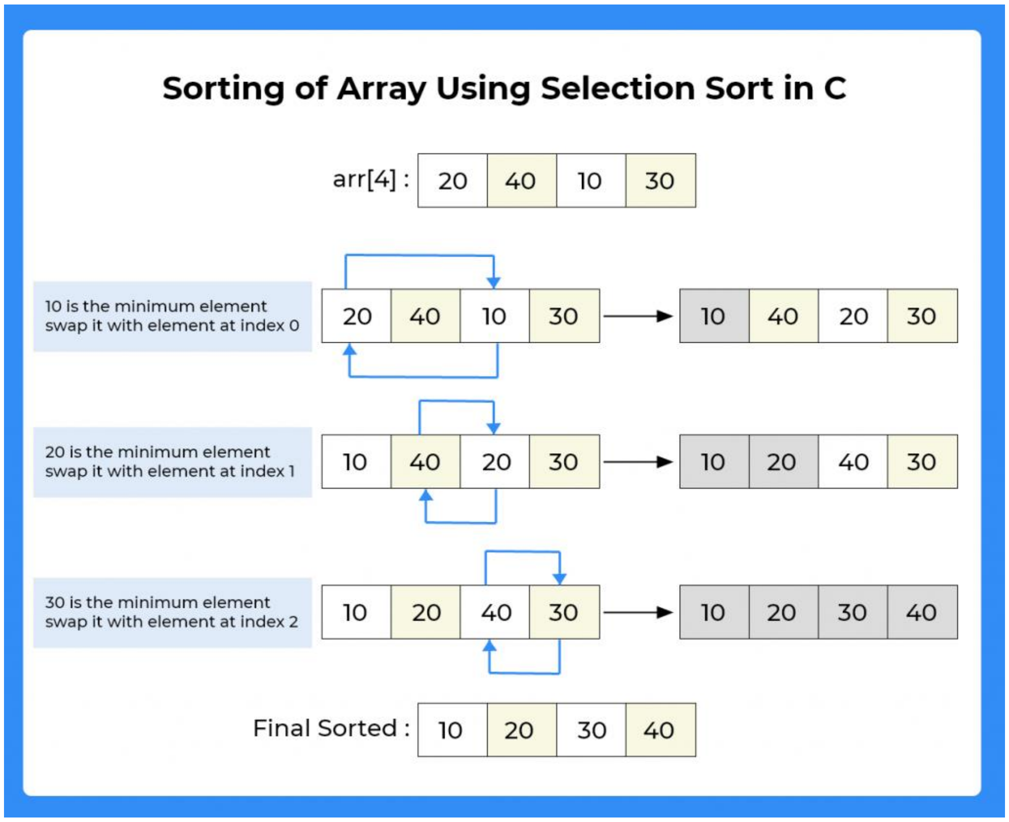
```
#include <stdio.h>
 #include <conio.h>


int main()
{
    int a[10000],i,n,key;

    printf("Enter size of the  array : ");
    scanf("%d", &n);
    printf("Enter elements in array : ");
    for(i=0; i<n; i++)
    {
        scanf("%d",&a[i]);
    }
     printf("Enter the key : ");
    scanf("%d", &key);

    for(i=0; i<n; i++)
    {
        if(a[i]==key)
        {
                                        printf("element found ");
            return 0;
        }

    }

             printf("element  not  found");
}

OUTPUT
```

Sorting

Sorting is the process of arranging elements in a list or array in a specific order, typically in ascending or descending order. Sorting is a fundamental problem in computer science and has many applications in areas such as searching, data compression, and data analysis.

There are many sorting algorithms that can be used to sort an array. Some of the most popular algorithms include:

1. **Bubble Sort:** This is a simple sorting algorithm that repeatedly steps through the list, compares adjacent elements, and swaps them if they are in the wrong order.
2. **Selection Sort:** This algorithm sorts an array by repeatedly finding the minimum element from the unsorted part of the array and putting it at the beginning.
3. **Insertion Sort:** This algorithm builds the final sorted array one item at a time, by inserting each item in the correct position in the array.
4. **Merge Sort:** This algorithm divides the array into two halves, sorts each half separately, and then merges the two halves together to form a sorted array.
5. **Quick Sort:** This algorithm picks an element as a pivot and partitions the array around the pivot, such that elements smaller than the pivot are on one side and elements larger than the pivot are on the other side. It then recursively sorts the two sub-arrays.


Sorting of Array Using Selection Sort in C
arr[4] : 20 40 10 30
10 is the minimum element swap it with element at index 0
20 40 10 30
10 40 20 30
20 is the minimum element swap it with element at index 1
10 40 20 30
10 20 40 30
30 is the minimum element swap it with element at index 2
10 20 40 30
10 20 30 40
Final Sorted : 10 20 30 40

Algorithm :

- Take the size of the array from the user.
- Declare an array of given input size.
- Take the input of all elements of the array.
- Now run a for loop from 0 to size-1.
- And for every element check it from all the next elements to it. If the element is greater than swap that number.
- In this way the array will get sorted in ascending order.

```c
#include<stdio.h>
int main()
{

    int a[6]= {12,5,10,9,7,6};
    int temp;
    int i, j;

     printf("Before Sorting ");

    for(i=0; i<6; i++)
    {
        printf("%d ",a[i]);
    }

    for(i=0; i<6; i++)
    {
        for(j=i+1; j<6; j++) { if(a[i]>a[j])
            {
                temp = a[i];
                a[i] = a[j];
                a[j] = temp;
            }
        }
    }
     printf("\nAfter Sorting ");
    for(i=0; i<6; i++)
    {
        printf("%d ",a[i]);
    }
    return 0;
}
```

Traversing

Given an integer array of size **N**, the task is to traverse and print the elements in the array.

Examples:

Input: *arr[] = {2, -1, 5, 6, 0, -3}*
Output: *2 -1 5 6 0 -3*
Input: *arr[] = {4, 0, -2, -9, -7, 1}*
Output: *4 0 -2 -9 -7 1*

CODE

```
/ C program to traverse the array

#include <stdio.h>

// Function to traverse and print the array
void printArray(int* arr, int n)
{
    int i;

    printf("Array: ");
    for (i = 0; i < n; i++) {
        printf("%d ", arr[i]);
    }
    printf("\n");
}

// Driver program
int main()
{
    int arr[] = { 2, -1, 5, 6, 0, -3 };
    int n = sizeof(arr) / sizeof(arr[0]);

    printArray(arr, n);

    return 0;
}
```

PRACTICAL 1B

read the two array from the user and merge them and display the element in sorted order

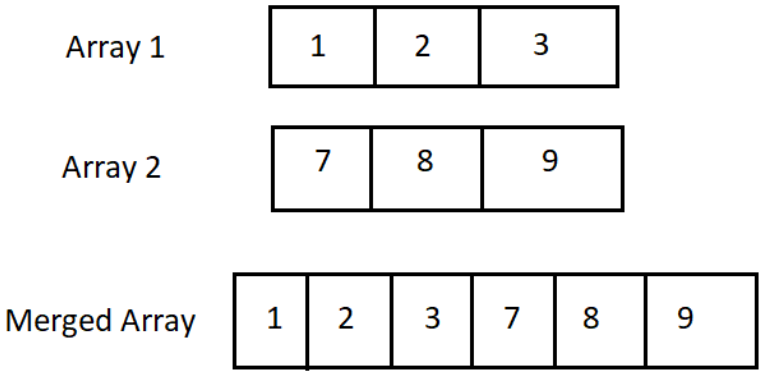
Merging two arrays means combining two separate arrays into one single array. For instance, if the first array consists of 3 elements and the second array consists of 5 elements then the resulting array consists of 8 elements. This resulting array is known as a merged array. Before moving forward, if you are not familiar with the concept of the array then, do check the article on Arrays in C.

We are given two sorted arrays and our task is to merge these two sorted arrays.

Input: First Array: 5 4 3 2 1

Second Array: 9 8 7 6 5

Output: Merged sorted Array: 9 8 7 6 5 5 4 3 2 1



CODE

```
#include <stdio.h>
int main()
{
   int n1,n2,n3;          //Array Size Declaration
int a[10000], b[10000], c[20000];

printf("Enter the size of first array: ");

scanf("%d",&n1);

printf("Enter the array elements: ");

for(int i = 0; i < n1; i++)

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

printf("Enter the size of second array: ");

   scanf("%d",&n2);

printf("Enter the array elements: ");

for(int i = 0; i < n2; i++)

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

n3 = n1 + n2;

for(int i = 0; i < n1; i++)

  c[i] = a[i];

for(int i = 0; i < n2; i++)

   c[i + n1] = b[i];


printf("The merged array: ");

for(int i = 0; i < n3; i++)

   printf("%d ", c[i]);       //Print the merged array


printf("\nFinal array after sorting: ");

for(int i = 0; i < n3; i++){

   int temp;

   for(int j = i + 1; j < n3; j++) {

      if(c[i] > c[j]) {

         temp = c[i];
```

```
                c[i] = c[j];
                c[j] = temp;
            }
        }
    }
    for(int i = 0; i < n3 ; i++)       //Print the sorted Array
        printf(" %d ",c[i]);
    return 0;
}
```

OUTPUT:

PRACTICAL 1C

Write a program to perform the matrix addition , multiplication and transpose operation.

1)Matrix addition

Matrices are widely used in various fields such as physics, engineering, and computer science. In C programming language, matrices are used to represent and manipulate multi-dimensional arrays of data. Here are a few examples of why we might need to use matrices in C:

CODE

To add two matrices in C programming language, you can use a nested for loop to iterate through each element of the matrices and add the corresponding elements together.

Here is an example of adding two matrices of size 3x3:

```
#include <stdio.h>
int main() {
    int a[3][3] = {{1, 2, 3}, {4, 5, 6}, {7, 8, 9}};
    int b[3][3] = {{9, 8, 7}, {6, 5, 4}, {3, 2, 1}};
    int c[3][3];

    int i, j;
    for (i = 0; i < 3; i++) {
```

```c
        for (j = 0; j < 3; j++) {
            c[i][j] = a[i][j] + b[i][j];
        }
    }
    printf("Result of addition: \n");
    for (i = 0; i < 3; i++) {
        for (j = 0; j < 3; j++) {
            printf("%d ", c[i][j]);
        }
        printf("\n");
    }

    return 0;
}
```