**1.Write a program in C to store elements in an array and print them**

**Code:**

```
#include <stdio.h>
int main()
{
   int a[10],n,i;
   printf("Enter the number of elements for the array :");
   scanf("%d",&n);
   for(i=0;i<n;i++)
   {
      printf("Enter the element a[%d]:",i);
      scanf("%d",&a[i]);
   }
   for(i=0;i<n;i++)
   {
      printf("%d\t",a[i]);
   }
   return 0;
}
```

**Output:**

```
Enter the number of elements for the array :5
Enter the element a[0]:1
Enter the element a[1]:2
Enter the element a[2]:3
Enter the element a[3]:4
Enter the element a[4]:5
1   2   3   4   5
```

**2.Write a program in C to count the frequency of each element of an array (Give input multiple times in repetition)**

**Test Data :**
**Input the number of elements to be stored in the array :3**
**Input 3 elements in the array :**
**element - 0 : 25**
**element - 1 : 12**
**element - 2 : 43**
**Expected Output :**
**The frequency of all elements of an array :**
**25 occurs 1 times**
**12 occurs 1 times**
**43 occurs 1 times**

**Code:**

```
#include <stdio.h>
int main()
{
   int a[10], n, i, temp, count[10];
   printf("Enter the number of elements for the array: ");
   scanf("%d", &n);
   for (i = 0; i < n; i++)
   {
      printf("Enter the element a[%d]: ", i);
      scanf("%d", &a[i]);
      count[i] = 0;
   }
   for (i = 0; i < n; i++)
   {
      printf("%d\t", a[i]);
   }
   for (i = 0; i < n; i++)
   {
      temp = a[i];
      for (int j = 0; j < n; j++)
      {
         if (temp == a[j])
         {
            count[i] = count[i] + 1;
         }
      }
   }
```

```
for (i = 0; i < n; i++)
   {
      int duplicate = 0;
      for (int j = 0; j < i; j++)
      {
         if (a[j] == a[i])
         {
            duplicate = 1;
            break;
         }
      }
      if (!duplicate)
      {
         printf("\nElement %d appears %d times", a[i], count[i]);
      }
   }
   return 0;
}
```

**Output:**

```
Enter the number of elements for the array: 5
Enter the element a[0]: 1
Enter the element a[1]: 1
Enter the element a[2]: 1
Enter the element a[3]: 2
Enter the element a[4]: 3
1   1   1   2   3
Element 1 appears 3 times
Element 2 appears 1 times
Element 3 appears 1 times
```

**3.Write a program in C to read n number of values in an array and display them in reverse order.**

**Test Data :**
**Input the number of elements to store in the array :3 Input 3 number of elements in the array : element - 0 : 2**
**element - 1 : 5**
**element - 2 : 7**

**Expected Output :**
**The values store into the array are :**
**2 5 7**
**The values store into the array in reverse are : 7 5 2**

**Code:**

```
#include <stdio.h>
int main()
{
   int a[10],n,i;
   printf("Enter the number of elements for the array :");
   scanf("%d",&n);
   for(i=0;i<n;i++)
   {
      printf("Enter the element a[%d]:",i);
      scanf("%d",&a[i]);
   }
   for(i=n-1;i>=0;i--)
   {
      printf("%d\t",a[i]);
   }
   return 0;
}
```

**Output:**

```
Enter the number of elements for the array :5
Enter the element a[0]:1
Enter the element a[1]:2
Enter the element a[2]:3
Enter the element a[3]:4
Enter the element a[4]:5
5   4   3   2   1
```

**4.Write a program in C to find a pair with given sum in the array.**

**Code:**

```
#include <stdio.h>
int main()
{
   int a[15],n,i,j,sum;
   printf("Enter the number of elements for the array :");
   scanf("%d",&n);
   for(i=0;i<n;i++)
   {
      printf("Enter the element a[%d]:",i);
      scanf("%d",&a[i]);
   }
   for(i=0;i<n;i++)
   {
      printf("%d\t",a[i]);
   }
   printf("\nEnter the sum :");
   scanf("%d",&sum);
   printf("The pairs that add up to %d:",sum);
   for(i=0;i<n;i++)
   {
      for(j=i+1;j<n;j++)
      {
         if((a[i]+a[j])==sum)
         {
            printf("(%d,%d)",a[i],a[j]);
         }
      }
   }
   return 0;
}
```

**Output:**

```
Enter the number of elements for the array :10
Enter the element a[0]:1
Enter the element a[1]:2
Enter the element a[2]:3
Enter the element a[3]:4
Enter the element a[4]:5
Enter the element a[5]:6
Enter the element a[6]:7
Enter the element a[7]:8
Enter the element a[8]:9
Enter the element a[9]:10
1   2   3   4   5   6   7   8   9   10
Enter the sum :10
The pairs that add up to 10:(1,9)(2,8)(3,7)(4,6)
```

**5.Write a program in C to find the largest sum of contiguous subarrays in an array**

**Code:**

```
#include <stdio.h>
int submax(int a[], int size)
{
   int current_sum = 0, maximum_sum = 0;
   for (int i = 0; i < size; i++)
   {
      current_sum = current_sum + a[i];
      if (current_sum > maximum_sum)
         maximum_sum = current_sum;
      if (current_sum < 0)
         current_sum = 0;
   }
   return maximum_sum;
}
int main()
{
   int a[15], n, i, lsum;
   printf("Enter the number of elements for the array: ");
   scanf("%d", &n);
   for(i = 0; i < n; i++)
   {
      printf("Enter the element a[%d]: ", i);
      scanf("%d", &a[i]);
   }

   lsum = submax(a, n);
   printf("Sum of the Largest Contiguous Sub-Array: %d", lsum);
   return 0;
}
```

**Output:**

```
Enter the number of elements for the array: 10
Enter the element a[0]: 1
Enter the element a[1]: 2
Enter the element a[2]: 3
Enter the element a[3]: 1
Enter the element a[4]: 5
Enter the element a[5]: 6
Enter the element a[6]: 7
Enter the element a[7]: 8
Enter the element a[8]: 1
Enter the element a[9]: 8
Sum of the Largest Contiguous Sub-Array: 42
```

**6. Write a program in C to find the missing number in array. There are no duplicates in the list**

**Code:**

```
#include <stdio.h>
int find(int a[], int n)
{
   int total = (n + 1) * (n + 2) / 2;
   for (int i = 0; i < n; i++)
   {
      total=total-a[i];
   }
   return total;
}
int main()
{
   int n;
   printf("Enter the number of elements in the array: ");
   scanf("%d", &n);
   int a[n];
   printf("Enter the elements of the array (without duplicates):\n");
   for (int i = 0; i < n; i++)
   {
      scanf("%d", &a[i]);
   }
   int missing = find(a, n);
   printf("The missing number is: %d\n", missing);
   return 0;
}
```

**Output:**

```
Enter the number of elements in the array: 5
Enter the elements of the array (without duplicates):
1
3
4
5
6
The missing number is: 2
```

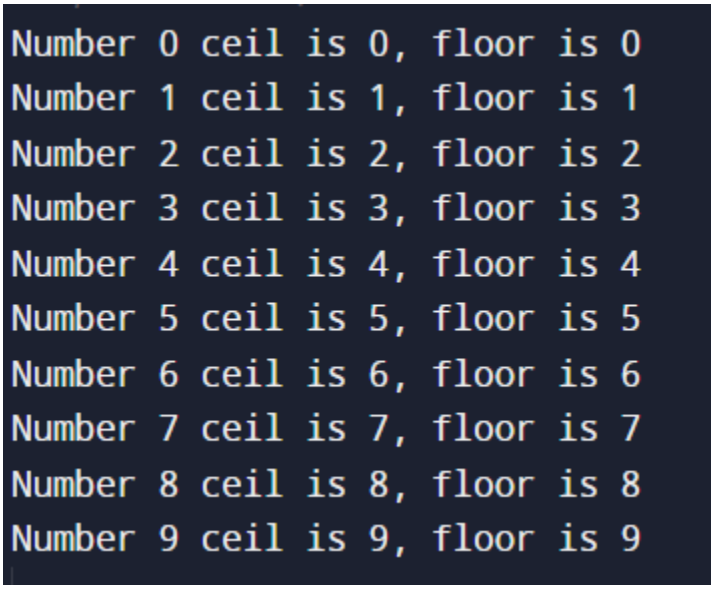
**7.Write a program in C to find the Floor and Ceiling of the number 0 to 10 from a sorted array**

**Code:**

```
#include <stdio.h>
int Ceil(int num[], int n, int x)
{
   int low = 0, high = n - 1, mid;
   int ceil = -1;
   while (low <= high)
   {
      mid = (low + high) / 2;
      if (num[mid] == x) {
         return num[mid];
      }
      else if (x < num[mid])
      {
         ceil = num[mid];
         high = mid - 1;
      }
      else
      {
         low = mid + 1;
      }
   }
   return ceil;
}
int Floor(int num[], int n, int x)
{
   int low = 0, high = n - 1, mid;
   int floor = -1;
   while (low <= high)
   {
      mid = (low + high) / 2;
      if (num[mid] == x) {
         return num[mid];
      }
      else if (x < num[mid]) {
         high = mid - 1;
      }
      else {
         floor = num[mid];
         low = mid + 1;
      }
   }
   return floor;
}
int main(void)
```

```
{
    int num[] = {0,1,2,3,4,5,6,7,8,9};
    int n = sizeof(num) / sizeof(num[0]);
    for (int i = 0; i < 10; i++)
    {
        printf("Number %d ", num[i]);
        printf("ceil is %d, ", Ceil(num, n, i));
        printf("floor is %d\n", Floor(num, n, i));
    }
    return 0;
}
```

**Output:**

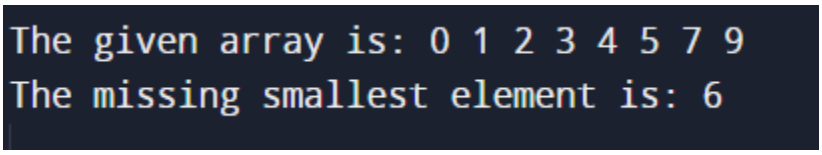


**8.Write a program in C to find the smallest missing element in a sorted array.**
**Expected Output :**
**The given array is : 0 1 3 4 5 6 7 9**
**The missing smallest element is: 2**

**Code:**

```
#include<stdio.h>
int missing(int a[], int n)
{
   int i;
   for(i = 0; i < n-1; i++) {
      if(a[i+1] - a[i] > 1) {
         return a[i] + 1;
      }
   }
   return -1;
}
int main()
{
   int a[] = {0, 1, 2, 3, 4, 5, 7, 9};
   int n = sizeof(a) / sizeof(a[0]);
   int missed = missing(a, n);
   printf("\nThe given array is: ");
   for(int i = 0; i < n; i++) {
      printf("%d ", a[i]);
   }
   if(missed!=-1)
   {
      printf("\nThe missing smallest element is: %d\n", missed);
   }
   else
   {
      printf("\nNo missing element found.\n");
   }
   return 0;
}
```

**Output:**



**9.Write a program to rotate an array of size n by d positions to the left.**
**For example, if the array is {1, 2, 3, 4, 5} and d is 2, the rotated array should be {3, 4, 5, 1, 2}.**

**Code:**

```
#include<stdio.h>
void rotate(int a[10],int d,int n)
{
   int temp[d];
   for (int i = 0; i < d; i++)
   {
      temp[i] = a[i];
   }
   for (int i = d; i < n; i++)
   {
      a[i - d] = a[i];
   }
   for (int i = 0; i < d; i++)
   {
      a[n - d + i] = temp[i];
   }
}
int main()
{
   int i;
   int a[] = {1, 2, 3, 4, 5};
   int d;
   int n = sizeof(a) / sizeof(a[0]);
   printf("Array before Rotation:\n");
   for (i = 0; i < n; i++)
   {
      printf("%d\t",a[i]);
   }
   printf("\nEnter the Value of 'd' (rotation factor):");
   scanf("%d",&d);
   rotate(a, d, n);
   printf("\nArray after Rotation :\n");
   for (i = 0; i < n; i++)
   {
      printf("%d\t", a[i]);
   }
   return 0;
}
```

**Output:**

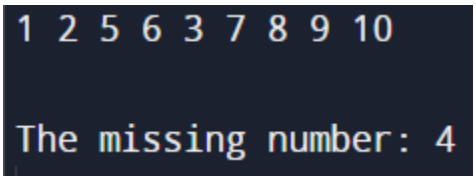
```
Array before Rotation:
1   2   3   4   5
Enter the Value of 'd' (rotation factor):1
Array after Rotation :
2   3   4   5   1
```

**10. Given an array containing n distinct numbers taken from the range 0 to n, find the missing number in the sequence. The array is missing exactly one number.**

**Code:**

```
#include<stdio.h>
int missing(int a[],int n)
{
    int totalsum= (n * (n + 1)) / 2;
    int sum = 0;
    for (int i = 0; i < n - 1; i++) {
        sum += a[i];
    }
    int missingno = totalsum - sum;
    return missingno;
}
int main()
{
    int a[] = {1, 2, 5, 6, 3, 7, 8, 9, 10}, i;
    int n = sizeof(a) / sizeof(a[0]) ;
    for (i = 0; i < n; i++)
    {
        printf("%d ",a[i]);
    }
    int missingno = missing(a,n);
    printf("\n\nThe missing number: %d\n", missingno);
    return 0;
}
```

**Output:**



**11. Given two sorted arrays, find the median of the combined array. This problem requires efficient merging of two sorted arrays and handling odd and even cases.**

**Code:**

```
#include<stdio.h>
int median(int a[], int b[], int n)
{
   int merged[2*n];
   int i=0, j=0, k=0;
   while (i < n && j < n)
   {
      if (a[i] <= b[j])
         merged[k++] = a[i++];
      else
         merged[k++] = b[j++];
   }
   while (i < n)
      merged[k++] = a[i++];
   while (j < n)
      merged[k++] = b[j++];
   if (k % 2 == 1)
      return merged[k / 2];
   else
      return (merged[(k - 1) / 2] + merged[k / 2]) / 2;
}
int main()
{
   int i,n;
   printf("Enter the size of the array A and B:");
   scanf("%d", &n);
   int a[n], b[n];
   printf("Enter the elements of the array A:");
   for (int i = 0; i < n; i++)
   {
      scanf("%d", &a[i]);
   }
   printf("Enter the elements of the array B:");
   for (int i = 0; i < n; i++)
   {
      scanf("%d", &b[i]);
   }
   int med = median(a, b, n);
   printf("Median: %d\n", med);
   return 0;
}
```

**Output:**

```
Enter the size of the array A and B:4
Enter the elements of the array A:1
3
5
7
Enter the elements of the array B:2
4
6
8
Median: 4
```

**12. Implement a program to represent a sparse matrix (a matrix with a majority of its elements being zero) using a structure and pointers.**

**Code:**

```
#include<stdio.h>
#include<stdlib.h>
struct sparsematrix
{
   int m;
   int n;
   int data;
   struct sparsematrix* next;
};
struct sparsematrix* build(int row, int col, int value)
{
   struct sparsematrix* element = (struct sparsematrix*)malloc(sizeof(struct sparsematrix));
   if (element == NULL)
   {
      printf("Memory allocation failed.\n");
      exit(1);
   }
   element->m = row;
   element->n = col;
   element->data = value;
   element->next = NULL;
   return element;
}
void show(struct sparsematrix* matrix)
{
   struct sparsematrix* temp = matrix;
   printf("\nSparse Matrix:\n");
   while (temp != NULL)
   {
      printf("(%d, %d) = %d\n", temp->m, temp->n, temp->data);
      temp = temp->next;
   }
}
void add(struct sparsematrix** matrix, int row, int col, int value)
{
   struct sparsematrix* newNode = build(row, col, value);
   if (*matrix == NULL)
   {
      *matrix = newNode;
   }
   else
   {
      struct sparsematrix* temp = *matrix;
      while (temp->next != NULL)
```

```
        {
            temp = temp->next;
        }
        temp->next = newNode;
    }
}
int main()
{
    int nrows, ncols, data;
    struct sparsematrix* sparseMatrix = NULL;
    printf("Enter the number of rows in the matrix: ");
    scanf("%d", &nrows);
    printf("Enter the number of columns in the matrix: ");
    scanf("%d", &ncols);
    printf("Enter the elements of the %d X %d matrix:\n",nrows,ncols);
    for (int i = 0; i < nrows; i++) {
        for (int j = 0; j < ncols; j++) {
            scanf("%d", &data);
            if (data != 0)
            {
                add(&sparseMatrix, i, j, data);
            }
        }
    }
    show(sparseMatrix);
    return 0;
}
```

**Output:**

```
Enter the number of rows in the matrix: 2
Enter the number of columns in the matrix: 2
Enter the elements of the 2 X 2 matrix:
1
2
3
4
Sparse Matrix:
(0, 0) = 1
(0, 1) = 2
(1, 0) = 3
(1, 1) = 4
```