CODE FOUNDATION

Arrays:

Arrays are the collection of **contiguous memory locations** which store the **same type of data**. Each location in an array is called a The numbers can repeat. Your task is to find the occurrence of each number.

In order to find the occurrence of each number you have to save them somewhere where you can access them easily. If I store the 5 numbers in different places(in 5 different variables) then it will be difn array element. Suppose you are given 5 numbers.ficult for me to access them. In arrays the numbers can be stored in contiguous memory locations which means if the first number is stored at memory location 200 then then the second element will be stored at memory location 204 and so on.

Note: If the array stores values of the same data type.

If the array stores integers and suppose the first element is at memory location 200, then the second array element will be at memory location 204 and the third element will be at memory location 208. Since each array element is of integer Type, therefore the difference between two neighbouring memory locations is 4. [Size of integer is 4 bytes]

Declaration:

int arr[5] declares an array of 5 elements of integer type.
char arr[5] declares an array of 5 elements of character type.

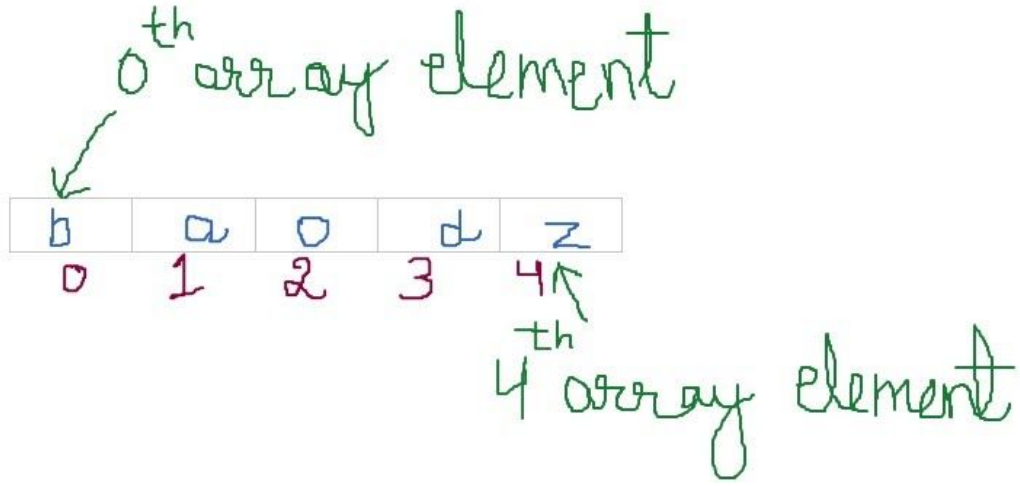
Example of an integer array. Here an integer array of size 5 is declared. We are assuming that the array stores integers 5,2,6,1,7. 5 is stored at memory location 200, 2 is stored at memory location 204 and so on. Note that at the time of array declaration, garbage values are placed at each memory location. We can initialise the array with our own values later on.

Ways of declaring arrays:

1. **int arr1[10];**
2. **int arr[] = { 10, 20, 30, 40 }**
3. **int arr[6] = { 10, 20, 30, 40 }**
4. **int n = 10;**
   int arr2[n];

Indexing in arrays:

In order to differentiate elements of an array, we provide different indices to them. We say that the first array element is at 0th index, second element is at index 1 and so on. Note the last element of array will be stored at index (size of array-1).

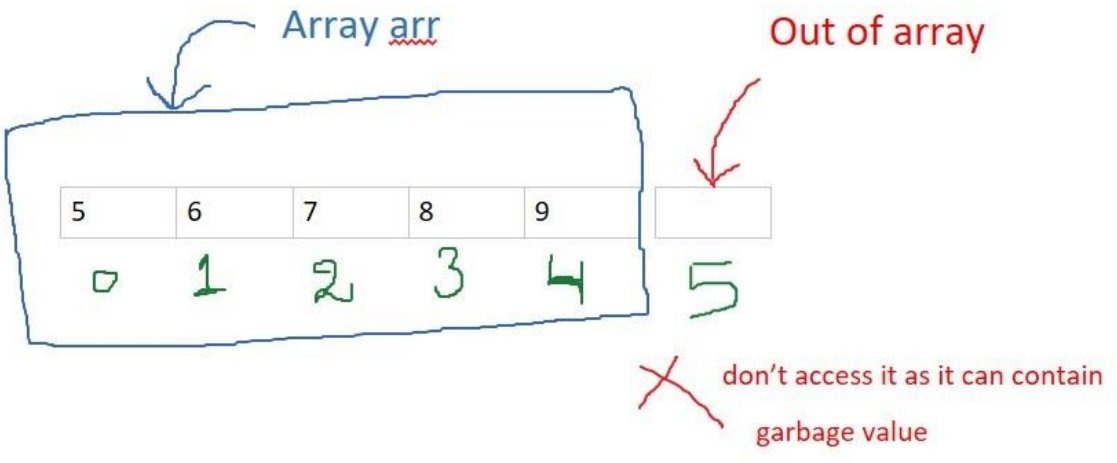


This is a character array of size 5 [there are 5 elements in it]. The indexing always starts with 0. Note that the last element is at index size-1 i.e 4.

Accessing elements in array :

How to store the values in an array? We will use the indexing to store values at different places in the array. Suppose you want to store 5 at first position in the array, you will write as **arr[0]=5**. To store 6 at the second position in the array, you will write as **arr[1]=6**.

Note that if you write arr[5] =8 it's wrong. Why? Here we are taking an example of an array with size 5. It will have the indexes 0,1,2,3,4. We try to access arr[5] means we are accessing a memory location outside the array and we are trying to change that memory location's value to 8. So this statement is wrong.



This is the pictorial depiction of the above example.
In order to print the value of suppose 3rd index element, we will write as **cout<<arr[3]**;
Similarly if we want the user to enter a value and store that vale into 3rd index of array arr, then we will write as **cin>>arr[3]**;

```
1. #include <iostream>
2. using namespace std;
3.
4. int main() {
5. int arr[5];
6. cout<<"Enter the array elements"<<endl;
7. for(int i=0;i<5;i++)
8. {
9. cin>>arr[i];
10.}
11.       return 0;
12.}
```

This code snippet takes input from the user. It takes the value for each element of the array from the user. Note the loop variable i starts from 0. It takes input from the user for

arr[0] i.e 0th element then i increments and i becomes 1. Then it takes input from the user for index 1. Loop variable increments till it is less than 5, i.e 4.

```
#include <iostream>
using namespace std;

int main() {
int arr[5];
for(int i=0;i<5;i++)
{
cin>>arr[i];
}
//printing array elements
for(int i=0;i<5;i++)
{
cout<<arr[i]<<" ";
}
      return 0;
}
```

In this code snippet we are taking input of an array of size 5 from the user and then after taking the input ,we are printing the contents of the array.

Question 1

What will be the output?

int arr[100];

cout << arr[0];

This will give us a garbage value. We have only declared an array of size 100. After declaration array also contains garbage values like variables. We need to initialise the array.

Question 2

How to access the 4th element of an array of size 10?

The 4th element of the array will be stored at index 3. Therefore **array[3]** is the correct answer.

Question 3

Can we declare an array like this : int arr[n] ?

If we take n from the user and then declare an array like arr[n], it may or may not give an error. For this we must declare and initialise n beforehand. But this is not considered as good practice.We should always declare constant sized arrays like arr[100], arr[5]. Declare an array to a sufficiently large value like 100,1000 as per your need so that you can work on a part of it.

How to find the maximum / minimum in an array?

We have to traverse all the elements of the array one by one. Take a variable max and initialise it with a sufficiently small value. Traverse the array elements. Check if the first element is greater than or equal to max. If it is, update the value of max . Now max contains the current maximum element. Now check for the next element of the array. Is it greater than max? If yes, then update max. Do this for all the remaining elements of the array.At the end max will contain the maximum value from the array.

Finding maximum element Link: ideone link

```
1.  #include <iostream>
2.  #include<climits>
3.  using namespace std;
4.
5.  int main() {
6.  int arr[100];
7.  int n;
8.  cout<<"Enter n:"<<endl;
9.  cin>>n;
10. for(int i=0;i<n; i++)
11. {
12.
13. cin>>arr[i];
14.
15. }
16.
17. int max=INT_MIN;
18. for(int i=0;i<n;i++)
19. {
20. if(arr[i]>max)
21. {
22. max=arr[i];
```

```
23. }
24. }
25. cout<<max<<endl;
26.         return 0;
27. }
```

Finding minimum element Link:[ideone link]

```
1. #include <iostream>
2. #include<climits>
3. using namespace std;
4.
5. int main() {
6. int arr[100];
7. int n;
8. cout<<"Enter n:"<<endl;
9. cin>>n;
10. for(int i=0;i<n; i++)
11. {
12.
13. cin>>arr[i];
14.
15. }
16.
17. int min=INT_MAX;
18. for(int i=0;i<n;i++)
19. {
20. if(arr[i]<min)
21. {
22. min=arr[i];
23. }
24. }
25. cout<<min<<endl;
26.         return 0;
27. }
28.
```

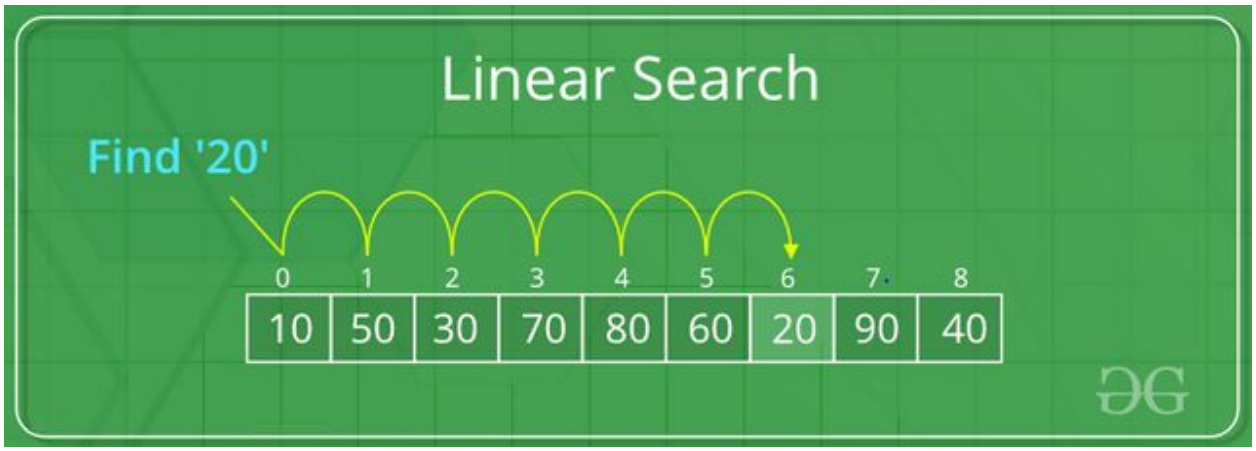
Searching an element in an array[Linear searching]

The simplest type of searching process is the sequential search/Linear search. In the sequential search, each element of the array is compared to the key.
(A **key** is a value that you are looking for in an array.)

In the order it appears in the array, until the first element matching the key is found. If you are looking for an element that is near the front of the array, the sequential search will find it quickly. The more data that must be searched, the longer it will take to find the data that matches the key using this process.

A simple approach is to do **linear search**, i.e

- Start from the leftmost element of arr[] and one by one compare x (key) with each element of arr[]
- If x matches with an element, print the index.
- If x doesn't match with any of elements, print not found.



Linear search program:
ProgramLink(SapphireEngine): https://sapphireengine.com/@/m40ch
ProgramLink(IDEONE): https://ideone.com/ckaus/linearsearch

```
#include <iostream>
using namespace std;

int main()
{
    int n = 9;  // size of the array
            int arr[n]; // declaration of the array
```

```
8.      cout << "Enter " << n << " elements in the array: " << endl;
9.      for (int i=0; i<n; i++)
10.         cin >> arr[i];
11.
12.     int key;
13.     cout << "Enter an element to search in the array: " << endl;
14.     cin >> key;
15.
16.     int indicate = 0, i;
17.     for (i=0; i<n; i++) {
18.         if (arr[i] == key) {
19.             indicate = 1;
20.             break;
21.         }
22.     }
23.
24.     if (indicate) cout << key << " is present at " << i+1 << " position in the array." << endl;
25.     else cout << key << " is not present in the array." << endl;
26.             return 0;
27. }
```

Standard Output

```
Enter 9 elements in the array: 10 50 30 70 80 60 20 90 40
Enter an element to search in the array: 20
20 is present at 7 position in the array.
```

Sum and Product of elements of array:

1. The user is initially asked to enter the size of the array and the value is stored in ‘n’.
2. The variables ‘sum’ and ‘pro’ are initialized as 0 and 1 respectively.
3. The user is asked to enter the array elements. Using a for loop, the elements are stored in the array ‘arr’.
4. Taking a for loop and initializing ‘i’ as 0, the sum and product are calculated and stored in sum and pro respectively.
5. i is incremented in every iteration. The loop continues till i is less than n.
6. The result is then printed.

Program:

sapphire engine link ideone link

```
#include <iostream>
using namespace std;

int main()
{
        int n=9;
    int a[n];
    cout << "Enter " << n << " elements in the array: " << endl;
    for (int i=0; i<n; i++)
        cin >> a[i];

    long prod=1, sum=0;
    for (int i=0; i<n; i++) {
        prod *= a[i];
        sum += a[i];
    }
    cout << "Sum = " << sum << endl;
    cout << "Product = " << prod << endl;
        return 0;
}
```

Standard Output

```
Enter 9 elements in the array: 1 2 3 4 5 6 7 8 9
Sum = 45
Product = 362880
```

Character arrays

We can only store a single character in a variable of char type. We don't have any method to store more than one character. Hence we need something to store a group of characters. We can use arrays of type char to do so. Character arrays work almost the same as integer arrays but they have some differences. We can input a character array directly without using loops. Note 1D character arrays are called **strings.**

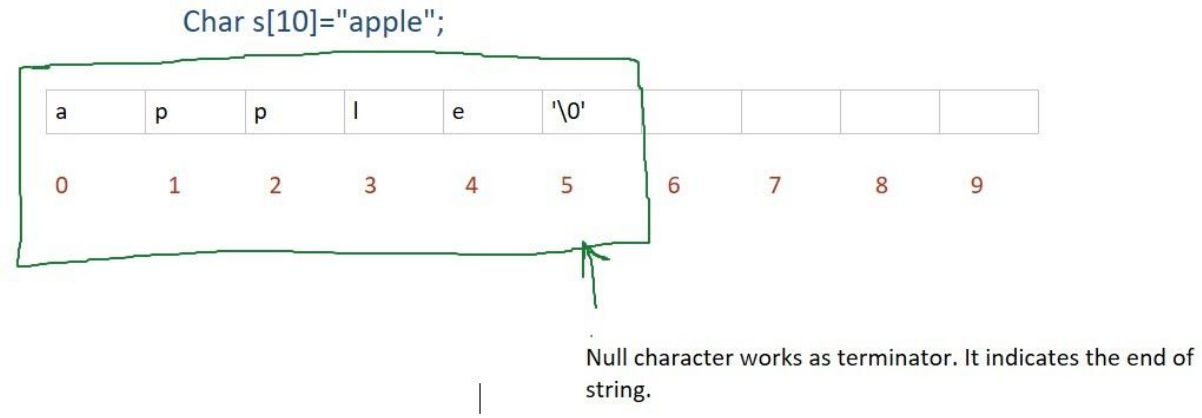
Eg **char s[10];**

We have declared a character array of size 10. Now we can directly take input from the user.

cin>>s;

See no need for loops!!

Another interesting thing in character arrays is that there is a null character in the end of the string. **Null character means '\0'**. It denotes the end of the string.



We can print the character array directly without using the loops.

Here cout<<s; will print apple on the screen.

Length of string:

To find the length of string traverse the whole character array till you find a null character and increment a variable count. If you encounter a null character it means you have reached the end of the string.

```
#include <iostream>
using namespace std;

int main() {
        // your code goes here

char s[20]="HAPPY";
int l=0;
for(int i=0;s[i]!='\0';i++)
{
```

```
11.
12.  l++;
13.  }
14.  cout<<l;
15.              return 0;
16.  }
```

Output:
5

Inbuilt functions for strings

To use these inbuilt functions you have to include cstring header file in your code.

#include<cstring>

1 strlen(string): This function is used to calculate the length of a string.

2 strcmp(string_1, string_2): This function is used to compare strings, whether they are equal or not. If it returns 0 it means the two strings are equal, else not equal.

3 strcpy(destination, source) : This function is used to copy the contents of one string to another.

Program:

sapphire engine link ideone link

```
1.  #include <iostream>
2.  #include <cstring>
3.  using namespace std;
4.
5.  int main()
6.  {
7.              char str1[25] = "I am a string";
8.     char str2[25] = "I am also a string";
9.     cout << "str1: " << str1 << endl;
10.    cout << "str2: " << str2 << endl;
11.    cout << "Length of str1 = " << strlen(str1) << endl;
12.    cout << "Length of str2 = " << strlen(str2) << endl;
13.    cout << "Are str1 and str2 equal? ";
14.    if (!strcmp(str1, str2))
15.       cout << "Yes" << endl;
16.    else
```

```
17.         cout << "No" << endl;
18.  
19.     char str1Copy[25];
20.     strcpy(str1Copy, str1);
21.     cout << "str1Copy: " << str1Copy << endl;
22.      cout << "Are str1 and str1Copy equal? ";
23.     if (!strcmp(str1, str1Copy))
24.         cout << "Yes" << endl;
25.     else
26.         cout << "No" << endl;
27.             return 0;
28. }
```

Standard Output

```
str1: I am a string
str2: I am also a string
Length of str1 = 13
Length of str2 = 18
Are str1 and str2 equal? No
str1Copy: I am a string
Are str1 and str1Copy equal? Yes
```

Problem 1:

Check whether the given string is palindrome or not :

A palindrome is a word or a string that reads the same backward and forward.

Example - **Civic , Madam , Racecar , Level etc.**

Approach towards problem Solution

1. The program takes a string and stores it.
2. The string is copied into another string from backwards.
3. If both the strings are equal, then the entered string is a palindrome.
4. Else it is not.
5. The result is printed.

6. Exit.

OR

Approach towards problem Solution

1. Take a string and store it in a character array.
2. Initiate a flag with true and a loop from i=0 to i<size/2 .
3. If both elements at ith and (size - i)th index the are equal, then continue iteration.
4. Else break the loop with a false flag signal.
5. Check the flag and accordingly print your answer.
6. Exit.

```
#include <iostream>
using namespace std;

int main() {
        // your code goes here
char str[100];
cin>>str;
int c=0,count=0,flag=0;
while(str[c]!='\0')
{
  c++;
  count++;
}

  int i=0,j=count-1;
  while(i<=j)
  {
    if(str[i]!=str[j])
    {
      cout<<"false";
      flag=1;
      break;
    }
    i++;
    j--;
  }
 if(flag==0)
  cout<<"true";

        return 0;
}
```

Problem 2

Print the highest occurring character in the given string

Traverse the string and find the occurrence of all the characters and store their occurrence in an array. Also check if the occurrence is greater than a maximum value or not. If it is greater then update the maximum to the new value.

```
#include<bits/stdc++.h>
using namespace std;

int main(){
    char input[1000];
    cin>>input;
    int count[256]={0};
    int maxi=INT_MIN,maxi_ind;
    int i=0;
    char ans;
   while(i<strlen(input))
   {
      count[input[i]]++;
       if(count[input[i]]>maxi)
     {
        maxi_ind=i;
        maxi=count[input[i]];
          ans=input[i];
     }
      i++;

   }


    cout<<(ans);
}
```

2D Arrays

In C++, we can have multi dimensional arrays. We can think of multidimensional arrays like an array of arrays. Data in multidimensional arrays are stored in tabular form (in row major order). Total number of elements that can be stored in a multidimensional array can be calculated by multiplying the size of all the dimensions.
Example arr[10][5] can store 10x5 elements i.e 50 elements.

Simplest multidimensional array is a 2D array. Example of a 2D array is arr[5][5].

int arr[5][6]

	0	1	2	3	4	5
0	6	7	3	8	2	7
1	8	1	2	4	0	9
2	1	4	7	89	45	3
3	87	56	23	676	13	1
4	4	80	4	12	65	98

Here is the pictorial representation of a 2D array.
Elements in two-dimensional arrays are commonly referred by x[i][j] where i is the row number and 'j' is the column number.You can use other variables also.

A two – dimensional array can be seen as a table with ‘x’ rows and ‘y’ columns where the row number ranges from 0 to (x-1) and column number ranges from 0 to (y-1).A two dimensional array ‘x’ with 3 rows and 3 columns is shown below:

	Column 0	Column 1	Column 2
Row 0	x[0][0]	x[0][1]	x[0][2]
Row 1	x[1][0]	x[1][1]	x[1][2]
Row 2	x[2][0]	x[2][1]	x[2][2]

Initializing 2D array

int x[3][4] = {0, 1 ,2 ,3 ,4 , 5 , 6 , 7 , 8 , 9 , 10 , 11}

The above array has 3 rows and 4 columns. The elements in the braces from left to right are stored in the table also from left to right. The elements will be filled in the array in the order, first 4 elements from the left in first row, next 4 elements in second row and so on.

How to take a 2D array as input from the user and print it?

Here we are taking a 3x3 2D array input from the user and printing it.

ideone link sapphire engine link

```
1.  #include <iostream>
2.  using namespace std;
3.
4.  int main() {
```

```
5.
6.  int arr[10][10];
7.  int m=3,n=3;
8.
9.  for(int i=0;i<m;i++)
10.  {
11.      for(int j=0;j<n;j++)
12.      {
13.          cin>>arr[i][j];
14.      }
15.  }
16. for(int i=0;i<m;i++)
17.  {
18.      for(int j=0;j<n;j++)
19.      {
20.          cout<<arr[i][j]<<" ";
21.
22.      }cout<<endl;
23.  }
24.
25.              return 0;
26. }
```

Problem 3

For a given two-dimensional integer array/list of size (N x M), print the array/list in a sine wave order, i.e, print the first column top to bottom, next column bottom to top and so on.

ideone link
INPUT

```
3 3
1 2 3
4 5 6
7 8 9
```

OUTPUT

```
1 4 7 8 5 2 3 6 9
```

```
#include <iostream>
using namespace std;

int main() {

int row,col;
cin>>row>>col;
int input[row][col];
for(int i=0;i<row;i++)
 {
    for(int j=0;j<col;j++)
    {
       cin>>input[i][j];
    }
}

for(int j=0;j<col;j++)
{
  if(j%2!=0)
  {
     for(int i=row-1;i>=0;i--)
     {
        cout<<input[i][j]<<" ";
     }
  }
  else{
  for(int i=0;i<row;i++)
  {
     cout<<input[i][j]<<" ";
  }
  }
}
            return 0;}
```

Extra Question 1

https://codeforces.com/problemset/problem/266/B

In this question we are given that there is a queue of n number of people. If at time x a boy is standing on ith position and a girl is standing on (i+1)th position then the boy feels awkward and let's the girl on his position. It means at time x+1, girl will be at ith position and the boy will be at (i+1)th position. We have to print the queue after t seconds.

We can iterate over the whole loop every time to check for all the positions where a boy is standing immediately before a girl. If we encounter such a position then we will swap them. We will do this for each second. Then after t seconds we can display the modified queue.

Program link:https://sapphireengine.com/@/gme95u

Practice questions

1 Given a string S (that can contain multiple words), you need to find the word which has minimum length. If multiple words are of same length, then the answer will be the first minimum length word in the string.
Words are separated by single space only. [Note that getline function is used to take input of a string consisting of spaces in C++ .
Example:

```
cin.getline(name of string,length); ]
```

2 https://codeforces.com/problemset/problem/266/A

3 Given a string S, the task is to remove all the duplicates in the given string.
Example input: geeksforgeeks, Output:geksfor