Char Arrays

Arrays of characters is known as Character Arrays.

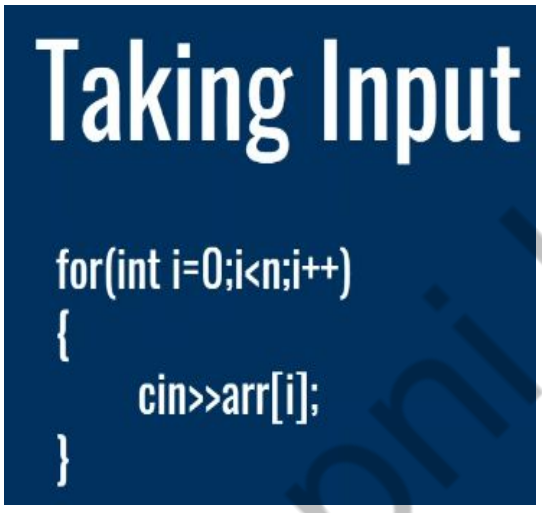
Declaration

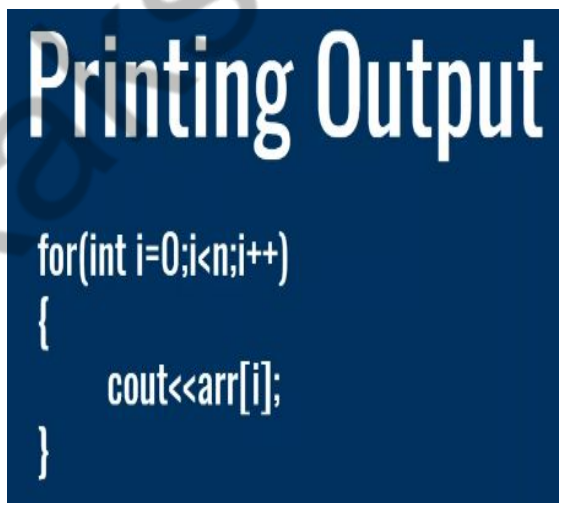
To declare a character array of n size, we do

char arr[n+1];

Note: We declare an array of n+1 as 0 to n-1 indices store the actual string and nth character stores '\0' (null character).

Taking input and **Printing output**





We can also directly take input if there are no spaces between the characters in the word

cin >> arr;

In the similar way, we can output the character array

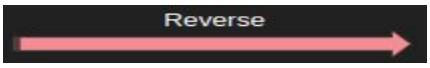
cout << arr;

Important Questions

1. <u>Check if a given character array is a palindrome or not.</u>

Palindrome: Given a string s, on reversing the string we get the same string we call that string is a palindrome.

For example:

racecar 

racecar

Algorithm:

1) Let the length of the character array be n.

2) Keep a boolean variable ans to store the result and initialize it with true.

2) Iterate over the string and check if i^{th} character is equal to $(n-i-1)^{th}$, there can be 2 cases

   a) If equal, then do nothing
   b) If unequal, then put ans = false

3) When the loop ends, if ans is true, then the string is palindrome else it is not a palindrome.

Code:

File Edit Selection Find View Goto Tools Project Preferences Help

soul.cpp | testing.cpp | temp.cpp

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

int main()
{
    #ifndef ONLINE_JUDGE
    freopen("input.txt", "r", stdin);
    freopen("output.txt", "w", stdout);
    #endif

    int n;
    cin >> n;

    char arr[n];
    cin >> arr;

    bool check = 1;
    for(int i=0; i<n; i++)
    {
        if(arr[i] != arr[n-1-i])
            check=0;
    }

    if(check)
        cout << arr <<" is a pallindrome" << endl;
    else
        cout << arr <<" is not a pallindrome" << endl;
}
```

input.txt

```
7
racecar
```

output.txt

```
racecar is a pallindrome
```

2. <u>Largest word in a sentence</u>

To input a complete sentence, we use the getline() function.

cin.getline(arr, n)

where arr is the character array and n is the total length of sentence.

Approach

1. Iterate over the sentence and keep variables currLen and maxLen which store the current length of the present word being iterated and the overall maximum length word's length.
2. Whenever we encounter a space during iteration, we will maximize our maxLen variable.

maxLen = max(maxLen, currLen)

Code

```
void solve()
{
    char arr[100] = "apple";
    int n;
    cin >> n;
    cin.ignore();
    cin.getline(arr, n);
    cin.ignore();

    int i=0;
    int currLen = 0;
    int maxLen = 0;
    int st = 0, maxSt = 0;
    while(i<n)
    {
        if(arr[i] == ' ' || arr[i] == '\0')
        {
            if(currLen > maxLen)
            {
                maxSt = st;
                maxLen = currLen;
            }

            st = i+1;
            currLen = 0;
        }
        else
            currLen++;

        if(arr[i] == '\0')
            break;
        i++;
    }

    cout << arr << endl;
    cout << maxLen << endl;

    for(int i=0; i<maxLen; i++)
        cout <<arr[maxSt+i];
    cout << endl;
}
```

```
10
do or die
```

output.txt

```
do or die
3
die
```