C++

LAUNCHPAD

CODING BLOCKS

Lecture-7

Arrays

- Character Arrays
- 2D Arrays

Prateek Narang

Doubts ?

Character Arrays!

Character Array Basics

- char str[100];
- char str[4] = { ‘A’, ‘B’, ‘C’, ‘D’};
- char str[] = {‘A’, ‘B’, ‘C’};
- char str[] = “Welcome”;
- char str[8] = “Welcome”;

CODING BLOCKS

So what are strings?

- In C/C++ we use a character array to simulate strings.
- By convention, a string is a sequence of characters followed by a null character.
- Null characters is a special character whose ascii value is 0 and its representation is '\0'
- In the previous slide example 4 and 5 are valid strings.

CODING BLOCKS

Printing a character array

- cout command treats characters address differently.
- If you give it any other type of address it will just print the address
- But if you give it an address of type character it will print characters byte by byte starting from that byte till it finds a byte which stores null character.
- It doesn't care about the allocated space.

CODING BLOCKS

Lets see it with some code.

Reading a string.

- We can read character by character from the screen and keep adding to an array till we find our delimiter which in most cases is '\n' and append 0 character to the end of the array.
- cin.getline

CODING BLOCKS

cin.getline

- cin.getline(char * BA, int max_space);
- cin.getline(char * BA, int max_space, char delimiter);

max_space is the available space starting from the passed address.

delimiter is the character which specifies the end of the string. By default it is '\n'.

cin.getline would automatically add '\0' at the end.

CODING BLOCKS

Since end of the string can be checked by looking for null character('\0') we don't need to pass number of elements to a function.

Lets do some problems!

- Calculate Length of the String
- Check if a string is palindrome or not

Time to Try ?

- Write a function which takes two strings A and B and appends B to A.
- Read N strings from a user and print the largest string.

CODING BLOCKS

Again – We can only initialize the array and not assign!
So if you want to update the string, you need do it character by character.

Always remember to append null character at the end of the string after any operation.

CODING BLOCKS

char * ptr VS char arr[]

- char array[] = “abc” sets the first four elements in array to ‘a’, ‘b’, ‘c’, and ‘\0’
- char *pointer = “abc” sets pointer to the address of the “abc” string (which may be stored in read-only memory and thus unchangeable)

CODING BLOCKS

Time to try ?

- Write a function to reverse a string.
- Given a string rotate it by n characters. e.g. if the string is CodingBlocks and n =3 then the output should be cksCodingBlo
- Write a function to check if two strings are permutations of each other.
- Write a program to print all substrings of a given string

CODING BLOCKS

2 D Arrays

CODING BLOCKS

2 D Arrays Declaration/Initialization

- int array1[2][3];
- int array2[2][3] = {{1,2,3}, {4,5,6}};
- Int array[][4] = {{1,2,3,4}, {4,5,6,7}, {8,9,10}};
- char array3[3][2] = {{'A','B'}, {'C','D'}, {'E','F'}};
- char array4[][4] = {"abc", "def", "efg", "hig"};

CODING BLOCKS

Accessing an array

- 2-D array can be visualized as a matrix with N rows and M Columns.
- First element is 0,0 and last is N-1, M-1
- To access jth element of ith row [considering i and j are 0 based] we can use arr[i][j] where arr is the name of the array.

CODING BLOCKS

Lets write some

- Read a matrix and find a number in it.
- Wave Print
- Spiral Print

CODING BLOCKS

Time to try?

- Write a program that determines which row or column in a 2d array of integers has the largest sum

How is it stored?

Depending on the architecture it could be either stored as:

- Column Major Form
- Row Major Form – Most common!

CODING BLOCKS

So what is arr[i][j]?

- We know that name of the array is address of first element.
- So when we are saying arr[i][j] its doing some calculation like *(arr + i * number of columns + j)
- Conceptually this is correct but actually this is wrong.

CODING BLOCKS

Lets look at 1-D array again

int arr[3] = {1,2,3};

- We know arr is an alias of address of first element i.e. arr == &arr[0]
- But what is &arr ? Initially its value is same as arr but lets just try to increment it by 1 and see. cout << &arr +1 << endl;
- This address is N*sizeof(data) far from the initial address where N is number of elements.
- So we can say &arr is also an address but its not address of one element but address of a complete row. We can say it's a pointer to array or a row pointer.

CODING BLOCKS

Lets see output of these statements

```
Int arr[][3] = {{1,2,3}, {4,5,6}, {7,8,9}};

cout <<arr << endl;
cout << arr+1 << endl;
cout << *(arr+1) << endl;
cout <<arr[0] << endl;
cout  <<*(arr[0]) << endl;
cout << &arr[0][0] << endl;
cout << arr[0] + 1 << endl;
cout << (&arr[0][0]) + 1 << endl;
cout << arr+1 << endl;
cout << &arr[1][0] << endl;
```

CODING BLOCKS

So what is actually arr[i][j]

- For a 1-D array arr[i] is smilar to *(arr+i)
- Similarly for 2-D array arr[i][j] is actually *(*(arr+i)+j)
- Now name of the array is a row pointer or we can say it is pointer to an array pointing to first array of the 2D.
- Its value is same as &arr[0][0] but its behavior is not.
- So &arr for a 2-D array is matrix pointer or we can say it is a pointer to array of arrays pointing to the complete matrix.

CODING BLOCKS

So finally we can say for 2-D array

Int arr[4][5];

- arr is an alias of address of first row or we can say it is a pointer to array of 5 ints which is currently pointing to first array.
- arr[0] is an alias of address of first element of first row (&arr[0][0]) or we can say it is a pointer to first element of 0^{th} row.
- Similarly arr[i] is an alias of address of first element of ith row.(&arr[i][0])
- &arr is an alias of address of the complete matrix of size 4*5 elements or its is pointer to a 2D array

CODING BLOCKS

Declaring pointer to array

- int (*p)[5] – This creates an pointer variable p which points to array of 5 integers.
- int *p[5] is not the same as above. This means an array of integer pointers.
- Round Brackets are important.

CODING BLOCKS

Passing 2-D arrays into a function.

- Like in a 1-D array when we pass it to function we are passing pointer to an element.
- Similarly for a 2-D array we are passing pointer to an array of size – number of columns.
- So a function declaration could either look like
  - void accept2D(int arr[][5])
  - void accept2D(int (*arr)[5])

CODING BLOCKS

Array of strings!

- We simulated a string by a 1-D character array.
- Similarly we can simulate a list of strings by 2-D character array.
- char stringlist[10][100];
- Above can store max 10 strings each of maxlength 100.
- And each string can be accessed by strlinglist[i].

CODING BLOCKS

Initializing array of strings!

Lets see an example.

- Given a list of strings and word S. Check if S exists in the list or not.

Time to try?

- Write a program to create a matrix of alternate rectangles of O and X

  For N = 5;

  OOOOO

  OXXXO

  OXOXO

  OXXXO

  OOOOO

- Read N words and sort them lexicographically.

CODING BLOCKS

What is next class about?

- Recursion

CODING BLOCKS

CODING BLOCKS

Thank You!