C++

Session IV

<TECH SHUTTLE/>

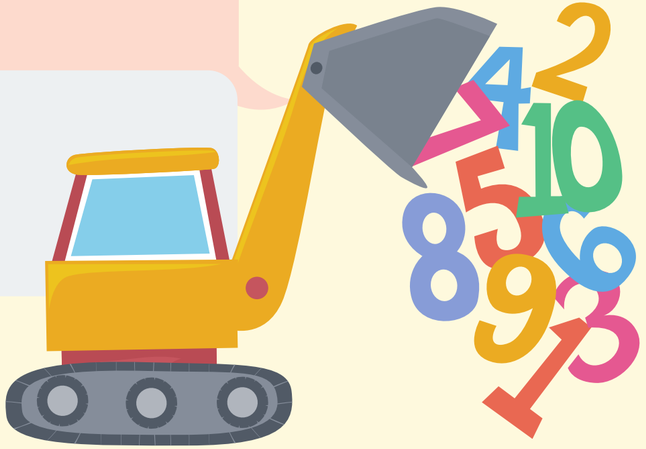
To be covered:

- ASCII Table
- Practice - Loops
- Pointers- Basics
- Arrays - 1-D

<TECH SHUTTLE/>

ASCII

Table



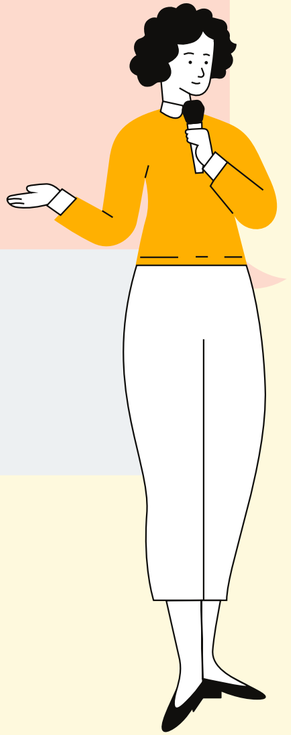
<TECH SHUTTLE/>

ASCII (American Standard Code for Information Interchange) is a standard character encoding used in telecommunication. They represent text in, telecommunications equipment, and devices. These include numbers, upper and lowercase English letters, functions, punctuation symbols, and some other symbols. In total, there are 256 ASCII characters.

Characters	Decimal
0-9	48-57
A-Z	65-90
a-z	97-122

Practice

Loops



<TECH SHUTTLE/>

QUESTION 1:

Find power of a number

Write a program to find x to the power n (i.e. x^n). Take x and n from the user. You need to print the answer.

Note : For this question, you can assume that 0 raised to the power of 0 is 1

Sample Input 1 :

```
3 4
```

Sample Output 1 :

```
81
```

QUESTION 2 :

Print the following pattern for the given N number of rows.

Pattern for N = 4

```
*
**
***
****
```

Sample Input 1:

```
5
```

Sample Output 1:

```
*
**
***
****
*****
```

QUESTION 3 :

Print the following pattern for the given N number of rows.

Sample Input 1:

```
5
```

Sample Output 1:

```
A
BC
CDE
DEFG
EFGHI
```

QUESTION 4 :

Print the following pattern for the given N number of rows.

Pattern for N = 4

```
···1
··12
·123
1234
```

The dots represent spaces.

QUESTION 5:

Print the following pattern

Pattern for N = 4

```
···*
··***
·*****
*******
```

The dots represent spaces.

Pointers

Basics

<TECH SHUTTLE/>

Symbol Table

Symbol Table is an important data structure created and maintained by the compiler to keep track of semantics of variable

Items stored in Symbol table:

- Variable names and constants
- Procedure and function names
- Literal constants and strings
- Compiler generated temporaries
- Labels in source languages

Reference Variable

- When a variable is declared as a reference, it becomes an alternative name for an existing variable.
- A variable can be declared as a reference by putting ‘&’ in the declaration.
- Declaration

  data_type & variable_name2 = variable_name1;

- 'Address of' Operator

What are Pointers

- Pointers are variables that store the address of other variables.
- Declaration

  *data_type * pointer_name = & variable_name;*

- Size of pointers is 4 or 8 depending on the machine and compiler

Why are Pointers required?

- To say in layman language, they are useful to use a particular variable out of its scope.

**Address are stored in 'Hexa-Decimal'*

- **De-reference Operator (*)**

  They are used to return the value of the variable pointed by the pointer.

- **NULL Pointer**

  To initialize a pointer variable when that pointer variable isn't assigned any valid memory address yet.

  ```
  data_type *  pointer_name  =  0;
  ```

Pointer Arithmatic

Let suppose we have an integer pointer p *for a variable* var.

var = 5;

p = 880;

- *p++;*

  p will become 884;

- *p--;*

  p will become 880;

- *p++;

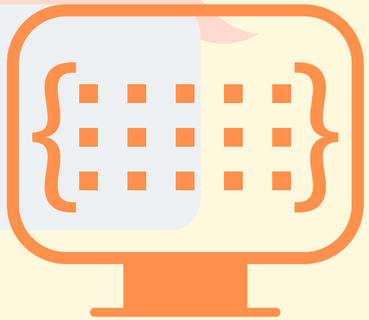
  var will become 6;

- *p--;

  var will become 5;

Arrays

1-d



<TECH SHUTTLE/>

Arrays

- Arrays are Linear Data Structures that store similar data items in contiguous memory locations.
- Elements can be accessed randomly using indices of an array.

Why do we need arrays?

- When dealing with many variables of similar types, it gets cumbersome to manage them. Hence, we need arrays.

Declaration & Initialisation

data_type array_name[no_of elements];

There are 3 ways to initialise:

1. **int arr[5] = {};**
2. **int arr[] = {1, 2, 3, 4, 5};**
3. **int arr[5] = {1, 2, 3, 4, 5};**

- Size of an array is equal to the size of the data type multiplied by the size of an array.

*size = n * sizeof(data_type)*

Storing of elements

- **Elements are stored in Arrays at certain indices.**

  0, 1, ..., (N-1)

  where N is the number of elements of the array

- Example:

40	55	63	17	22	68	89	97	89
0	1	2	3	4	5	6	7	8

<- Array Indices

Array Length = 9
First Index = 0
Last Index = 8

Arrays and Pointers

- In pointers, a new memory is created and then the address is stored in the pointer.
  While in the array, no new memory is created to store the address of the first element.

- In pointers, we can change the memory stored in the pointer.
  While in arrays, the array name pointer is unassignable, i.e. we can not change the address stored in the array name.

Accessing elements

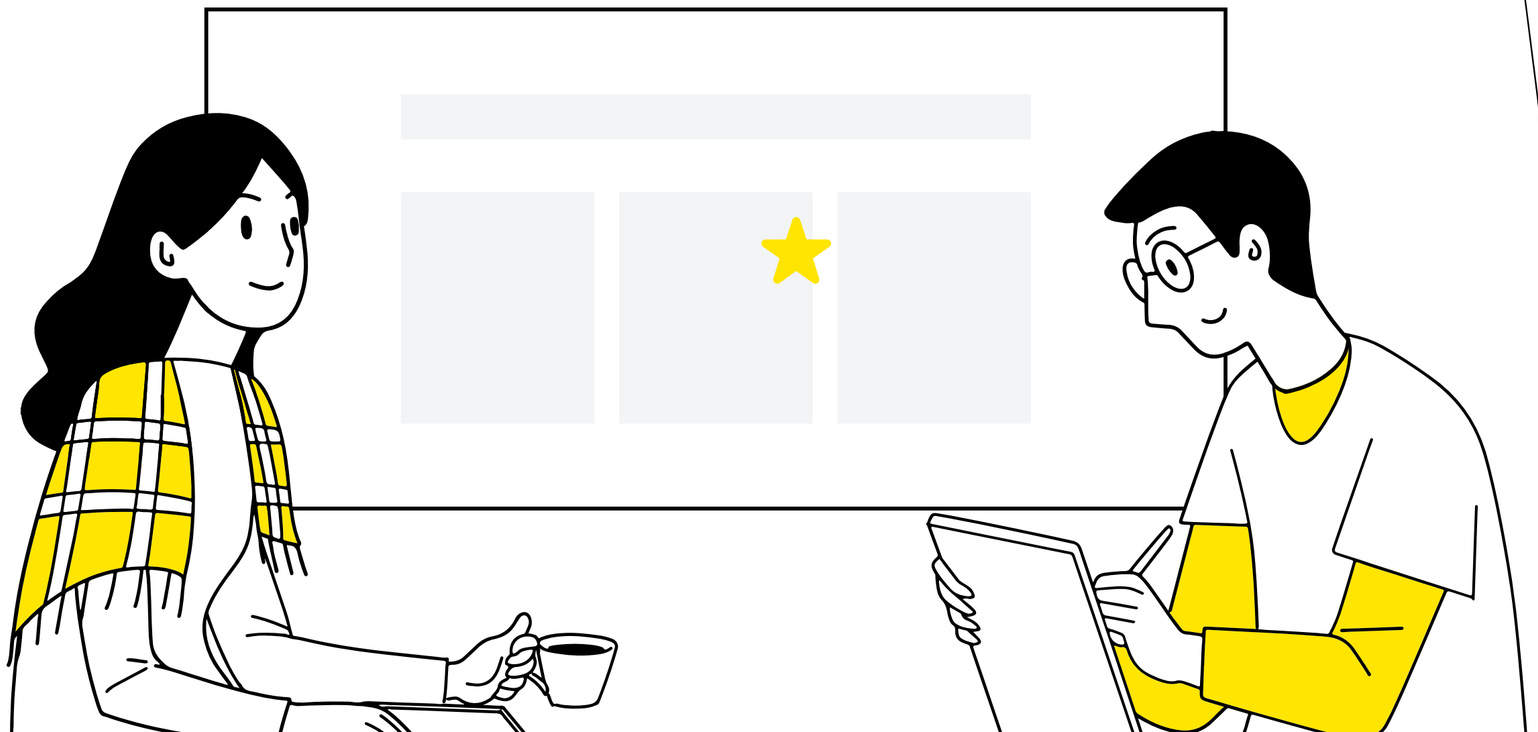
- **Elements can be directly accessed by using an index number.**
  - *array_name[index];*
  - *index[array_name];*
  - **(array_name + index);*

QUESTION TIME

Let's make the most out of our learning sessions.

<TECH SHUTTLE/>

Quiz Time!



The purpose of the quiz is not to shame or embarrass anyone, but to make sure everyone is on the same page.

See you in the next session Hope you had fun!
MARK