**C++ Onboarding**

1. **Refer the udemy course which can be accessed through mail id and password given below.**

→ mail id: [learning.yv@yantravision.com](mailto:learning.yv@yantravision.com)

Password: YvLearning@123

Link for the course: <https://www.udemy.com/course/beginning-c-plus-plus-programming/>

Finish the course from 1 to 13th Section.

1. **Finish the Exercises given Below:**

**1. 2D Arrays.**

→ Matrix multiplication program in C. Matrix Dimension and matrix values should be taken from arguments.

Ex: I/P: 1. Size of Matrix1: 2, 2

2. Size of Matrix2: 2, 2

3. Matrix1 Values: 10, 15, 20, 5

4. Matrix2 Values: 6, 14, 12, 20

O/P: 240, 440, 180, 380

**2. Pointer.**

→  **Write a C++ function to sort an array of ten integer values in ascending order.**

The function will accept two arguments-- a pointer that points to the array and the array size. The function returns a pointer that points to the sorted array.

**3. String operations(should not use predefined libraries).**

→ Concatenate

Ex: I/P: str1 -> “Programming”, str2 -> “is awesome.”

O/P: “Programming is awesome.”

→ Split the words from a sentence and arrange them in reverse order.

Ex: I/P: “Hello world”

O/P: “world Hell

**4. File Operations.**

→Open a file, Read the file. Every line changes every word’s first character to a capital letter and writes that data to another file.

Ex: I/P file: file1\_In.txt

O/P file: file1\_out.txt

**Data Structures:**

[**https://www.tutorialspoint.com/data\_structures\_algorithms/tree\_traversal.htm**](https://www.tutorialspoint.com/data_structures_algorithms/tree_traversal.htm)

**4. Linked list(One/two sorting algorithms).**

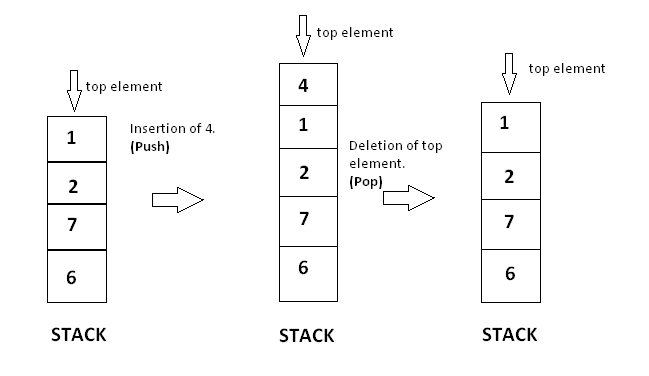
→ Write A Bubble Sorting algorithm using a linked list, Number of nodes should be created depending on the input data(Add the nodes dynamically and use structure pointers).

Ex: I/P array: 50, 34, 67, 2, 81, 46

O/P array: 2, 34, 46, 50, 67, 81

**6. Stack/Queue:**

→ Implement the Stack logic as shown in the picture below.(Last In First Out)



**7. Trees:**

**→** Write the Program to calculate the Pre-order, Post Order and In order of the nodes with value:

9, 4, 15, 6, 12, 17, 2

O/P: Pre-Order: 9, 4, 2, 6, 15, 12, 17

Post-Order: 2, 6, 4, 12, 17, 15, 9

In-Order: 2, 4, 6, 9, 12, 15, 17

**Multi-Threading:-**

**8. Multi-Threading (System Libs)**

**Links for multi threading intro:** [**https://www.youtube.com/watch?v=qPhP86HIXgg&t=32s**](https://www.youtube.com/watch?v=qPhP86HIXgg&t=32s)

[**https://www.geeksforgeeks.org/multithreading-c-2/**](https://www.geeksforgeeks.org/multithreading-c-2/)

**What is Deadlock:** [**https://en.wikipedia.org/wiki/Deadlock**](https://en.wikipedia.org/wiki/Deadlock)

**→ Deadlock:**

This example demonstrates how a deadlock can occur in multithreaded programs that use synchronization variables. In this example a thread is created that continually adds a value to a global variable. The thread uses a mutex lock to protect the global data.

The main thread creates the counter thread and then loops, waiting for user input. When the user presses the Return key, the main thread suspends the counter thread and then prints the value of the global variable. The main thread prints the value of the global variable under the protection of a mutex lock.

The problem arises in this example when the main thread suspends the counter thread while the counter thread is holding the mutex lock. After the main thread suspends the counter thread, it tries to lock the mutex variable. Since the mutex variable is already held by the counter thread, which is suspended, the main thread deadlocks.

This example may run fine for a while, as long as the counter thread just happens to be suspended when it is not holding the mutex lock. The example demonstrates how tricky some programming issues can be when you deal with threads.