

**TITLE:** Virtual Lab experiment on matrix multiplication

**AIM:** Virtual Lab experiment on recursion

**Batch: E1**

**Roll No.: 16010321011**

**Experiment / assignment / tutorial No. 7**

**Grade: AA / AB / BB / BC / CC / CD /DD**

**Signature of the Staff In-charge with date**

https://cse02-iiith.vlabs.ac.in/

https://cse02-iiith.vlabs.ac.in/exp/arrays/simulation.html

# Expected OUTCOME of Experiment:

**For 1D array**

**Please enter the values of array 4,9,1,5,2**

**The sorted array is 1,2,4,5,9**

**For 2D array**

**Please enter values of first 2\*2 matrix 7,12,13,2**

**Please enter the valus of second 2\*2 matrix 12,8,13,4**

**The matrix multiplication gives matrix**

**240 1**

**04**

**112**

**182**

**Books/ Journals/ Websites referred:**



1. Programming in C, second edition, Pradeep Dey and Manas Ghosh, Oxford University Press.
2. Programming in ANSI C, fifth edition, E Balagurusamy, Tata McGraw Hill.
3. Introduction to programming and problem solving , G. Michael Schneider

,Wiley India edition.

1. [**http://cse.iitkgp.ac.in/~rkumar/pds-vlab/**](http://cse.iitkgp.ac.in/~rkumar/pds-vlab/)

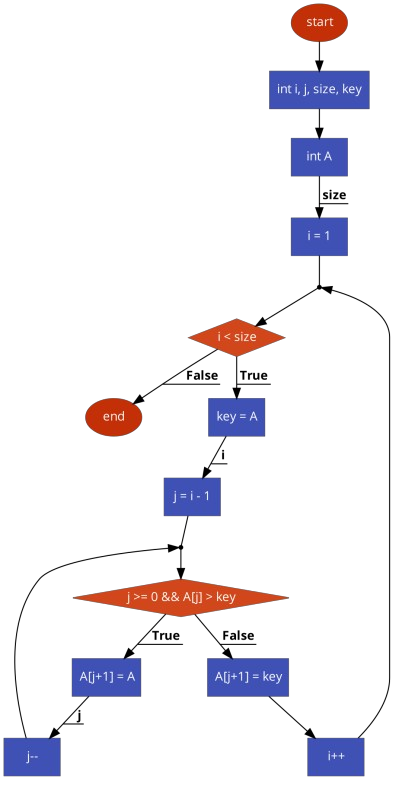
**Problem Definition:**

The Program implements 1-D and 2-D arrays.

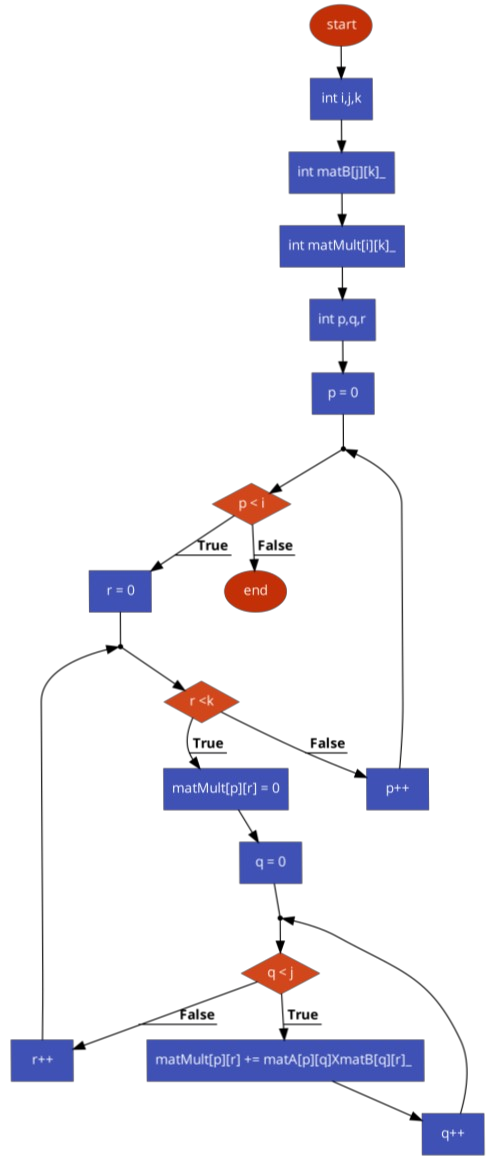
1-D array prompts the user to enter the size of the array and elements. It then sorts the elements.

* 1. array prompts the user to enter the order of two matrices and the elements. It then performs matrix multiplication.

**Flowchart: for 1D**

**For 2D**

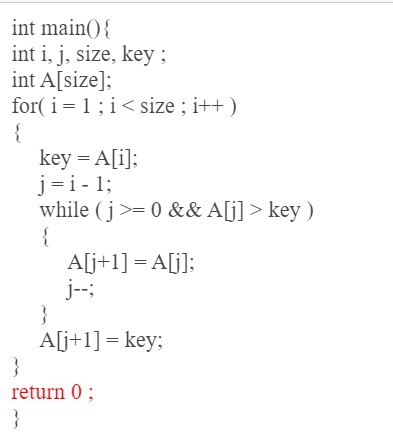
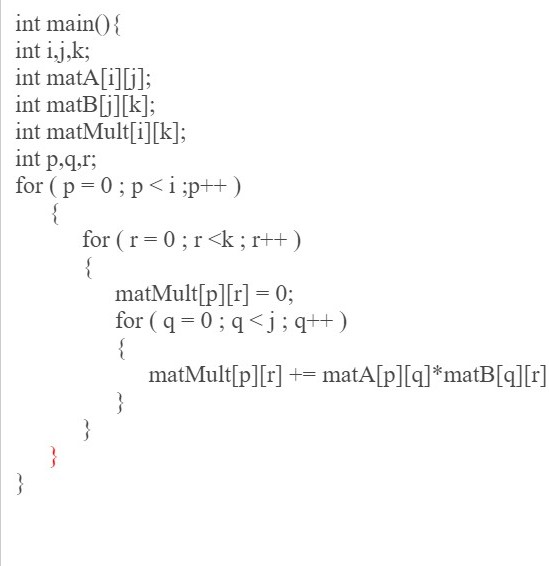






**Implementation details:**

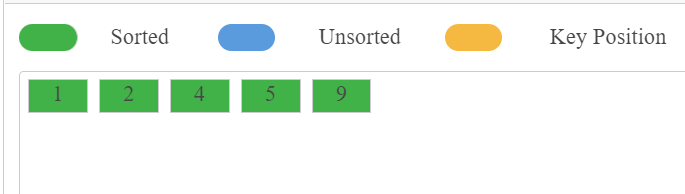
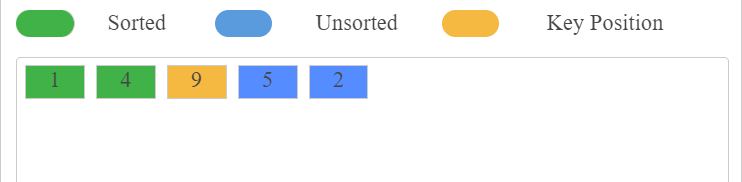
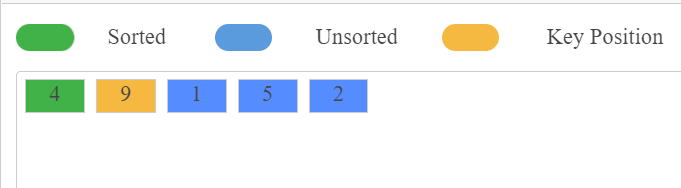
**For 1D For 2D**

# Output(s):

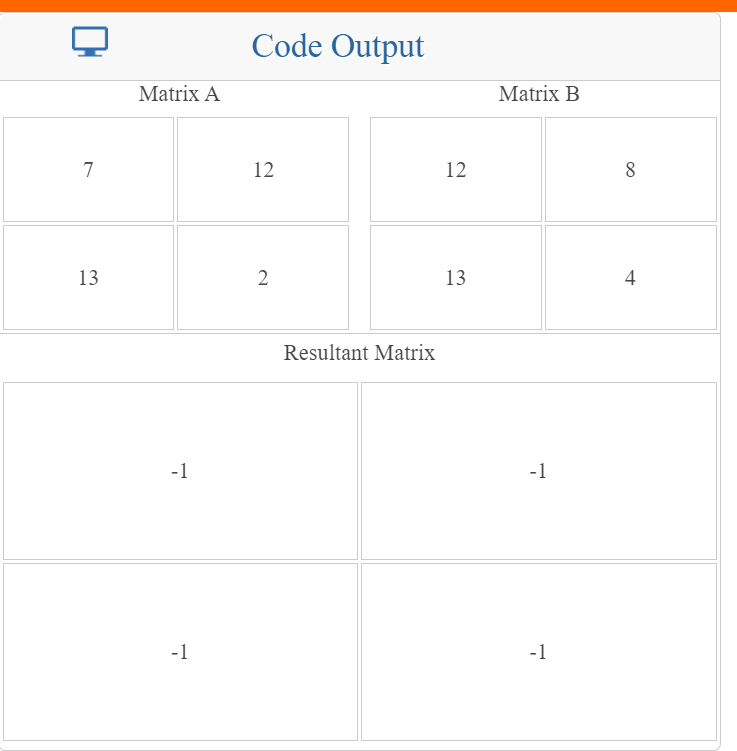
(Attach screenshots of the Output of Program Code implemented in Virtual Lab and Quiz attempted)

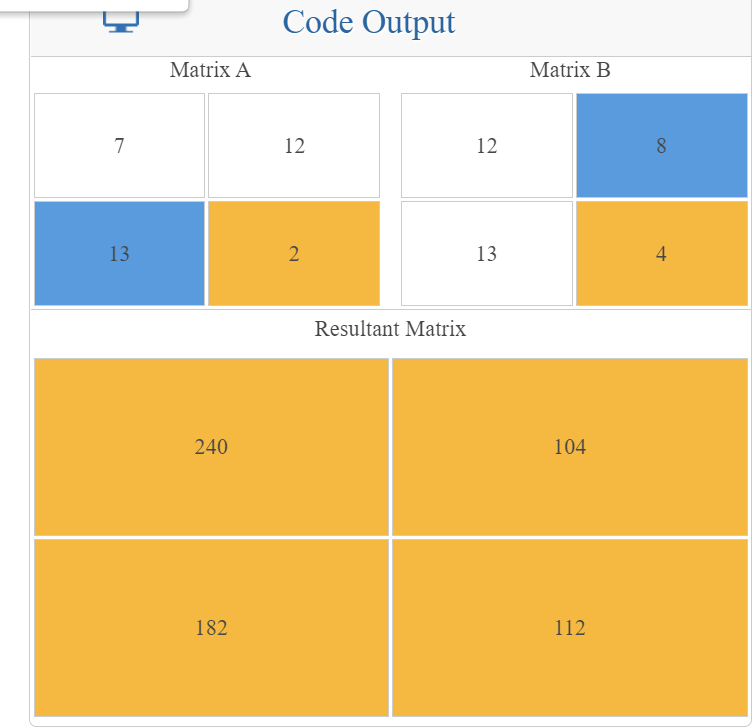
For 1D



For 2D



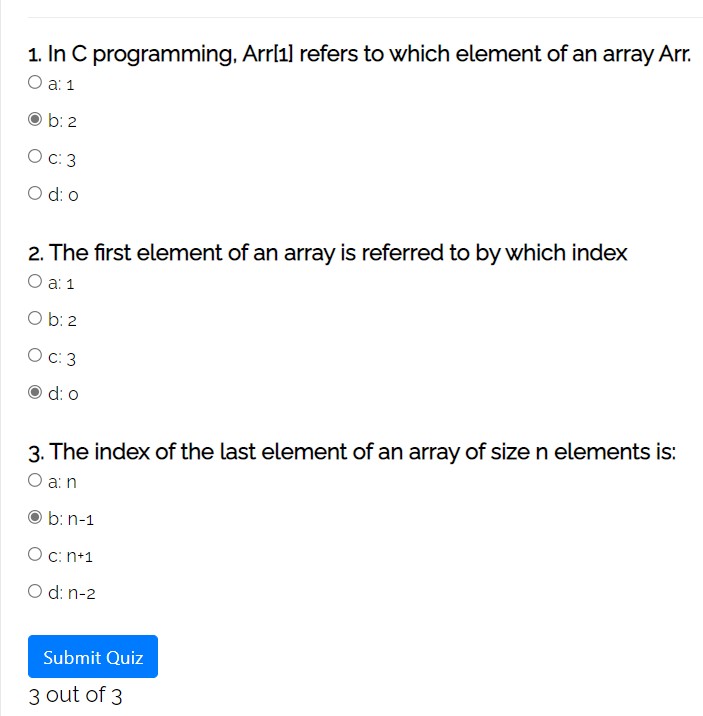


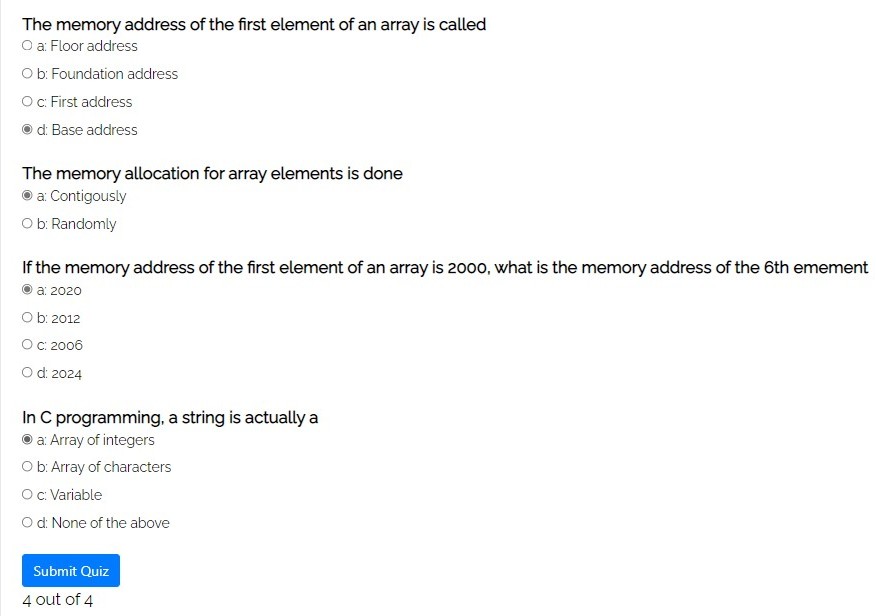




# Conclusion:

**Pre-test:**



**Post-test:**



**Post Lab Descriptive Questions**

* + 1. **Can we change the size of an array at run time? Why or why not?**

For providing memory on a stack the size of the memory should be known to the compiler during compile time. So that during run time that much memory can be set aside for the variable on the stack.

Hence we cannot decide the size of the array at run time as far as C language is concerned.

# Can we pass an array as an argument to a function?

There are two possible ways to do so, one by using call by value and other by using call by reference.

\*array as a parameter: Example: int sum (int arr[]);

\*pointers in the parameter list, to hold the base address of our array

:Example: int sum (int\* ptr);

# Date: Signature of faculty in-charge