National University
of computer and emerging sciences

Lab Task 02- Pointers

Problem 1:

Write a c++ program to accept five integer values from the keyboard. Store these five values in an array using pointer. Then print all the elements of an array in reverse order.
Note: use index number to return a value from an array

Problem 2:

Take an array of length n where all the numbers are nonnegative and unique. Find the element in the array possessing the highest value. Split the element into two parts where first part contains the next highest value in the array and second part hold the required additive entity to get the highest value. Return the array index value. Consider the negative numbers as well.
Input: 4 8 6 3 2 **Output:** 4 6 2 6 3 2
Note: use index number to return a value from an array

Problem 3:

Write a function that takes two arguments as input: a pointer to integer p and size s. You are required to allocate memory for an array with s elements of type int using new operator. As your next step, randomly initialize all the elements of the array and then update each element value with its square, i.e., replace each element value with its square.
Question for your thought why we are passing the pointer by reference into the function?

Problem 4:

Define and initialize five arrays of integer types each having 6 elements and an array of pointers p of size 5. Store the starting address of each array to array of pointers p. call the function and initialize the values of these five arrays randomly using for loop and p inside this function. Finally display the sum of each individual array.

Problem 5:

In the previous task we can consider p as a 2D matrix. However, in that case our number of rows (which were 5 as there were 5 arrays stored in p) and number of columns (which were 6 as each array had 6 elements) were fixed. In this question we will extend the previous question and make our number of columns variables. To do this, instead of using arrays of fixed size of 6 elements allocated on stack, we will make each array to have ncol (passed as argument to the function) elements and store them on heap using new operator. Once again store the starting address of each array to array of pointers p. Next, initialize the values of these five arrays randomly using for loop and p. Finally display the sum of each individual array.