**CMPSC122 In-Lab 3 – Pointers, Dynamic Arrays, and C-Strings**

Submit the solution files of Exercises online before the due date/time. Upload your file named InLab3.docx. You can use this handout and append your answer in between, please use a different color (preferably red) font for your answers for easy grading.

**Exercise 1.** (1 point) Determine the output of the following code segment. Please try to answer the questions with your knowledge before running these codes using the C++ compiler.

#include <iostream>

using namespace std;

void SomeFunc(int\* arr, int lowIndex, int highIndex);

int main()

{

int A[] = { 5, 10, 15, 20, 25, 30, 35, 40 };

int\* p = A;

cout << \*p << endl; **// Output: 5**

cout << p[3] << endl; **// Output: \_\_\_\_\_\_20\_\_\_\_\_\_\_**

cout << \*(p + 4) << endl; **// Output: \_\_\_\_\_\_25\_\_\_\_\_\_\_**

int\* q = p + 4;

cout << q[1] << endl; **// Output: \_\_\_\_\_\_\_30\_\_\_\_\_\_**

int val = 1;

for (int\* pi = A; pi < p + 7; pi++) {

\*pi = val;

val = val + 1;

}

cout << A[2] << endl; **// Output: \_\_\_\_\_\_3\_\_\_\_\_\_\_**

cout << \*(p + 3) << endl; **// Output: \_\_\_\_\_4\_\_\_\_\_\_\_\_**

int\* ptr = A + 4;

cout << \*ptr << endl; **// Output: 5**

SomeFunc(p+2, 1, 4);

for (int i = 0; i < 8; i++) {

cout << A[i] << " "; **// Output:** 1 2 3 8 10 12 14 40**\_\_\_\_\_\_**

}

cout << endl;

}

void SomeFunc(int\* arr, int lowIndex, int highIndex) {

for (int i = lowIndex; i <= highIndex; i++) {

arr[i] \*= 2;

}

}

**Exercise 2.** (1 point) Determine the output of the following code segment. Please try to answer the questions with your knowledge before running these codes using the C++ compiler.

#include <iostream>

#include <cstring>

using namespace std;

void SomeFunc(char \* cstr);

int main()

{

char str1[12] = "THISISATEST";

cout << str1 << endl; **// Output: THISISATEST**

cout << str1 + 4 << endl; **// Output: \_\_ISATEST\_**

str1[6] = '\0'; // Set to null byte

cout << str1 << endl; **// Output: \_\_THISIS\_\_\_\_\_\_\_\_**

char str2[] = { 'T', 'E', 'S', 'T', '\0', 'T', 'W', 'O', '\0', };

cout << strlen(str2) << endl; **// Output: \_\_\_\_\_\_\_\_\_\_4\_\_\_\_\_\_\_\_\_**

if (strcmp(str1 + 7, str2) == 0) {

cout << "Equal!" << endl;

}

else {

cout << "Not Equal!" << endl;

} **// Output: \_\_\_Equal!\_\_\_\_\_\_\_\_\_**

**// (Equal!/Not Equal!)**

SomeFunc(str1);

cout << str1 << endl; **// Output: \_\_\_\_\_\_HISIS\_\_\_\_**

}

void SomeFunc(char\* cstr) {

int len = strlen(cstr);

for (int i = 0; i < len; i++) {

cstr[i] = cstr[i + 1];

}

}

**Exercise 3.** (2 points) Quick Quiz (open book / open note).

1. (True or False) Let A an int type array. Then “cout << A;” and “cout << &A[0];” will give the same output to the screen. \_\_\_True\_\_
2. In C++, when passing an array as a parameter of a function call, the \_\_\_address\_\_\_\_\_\_\_\_ of the array is passed.
3. What is the name of the function from <cstring> that allows appending a cstring to the end of another one: \_\_\_\_\_\_\_strcat\_\_\_\_\_\_\_\_\_\_
4. (True or False) We can use .length() to obtain the length of a cstring. \_\_\_\_\_\_\_False\_\_\_\_\_
5. Write a statement to deallocate (delete) the memory that was allocated to the array dubArray with the statement

double \* dubArray = new double[100];

Your Answer (delete the memory of the dynamic array): \_\_\_delete[] dubArray;\_\_\_\_\_

**Exercise 4.** (6 points) Finish and test the following three functions in the skeleton code below. Upload your answers in a file named InLab3.cpp.

1. function **int\* append(int\*, int, int\*, int);** which accepts two arrays and returns a new array by appending the second array to the first array.
2. function **int\* merge(int\*, int, int\*, int);** which accepts two sorted arrays and returns a new merged sorted array.
3. function **void print(int\*, int, const char \*);** which accepts one array and prints the data in this array on-screen in the original order.

// Check your outputs for correctness.

// MUST test your code first before appending your code over here

**int main()**

**{**

**int arrayA[] = {11,33,55,77,99};** // use other values for more tests

**int arrayB[] = {22,44,66,88};**

**print(arrayA, 5, "Sorted array A: ");**

**print(arrayB, 4, "Sorted array B: ");**

**int\* arrayC = append(arrayA,5,arrayB,4);** // arrayC points to the appended array

**print(arrayC, 9, "Append B to A: ");**

**int\* arrayD = merge(arrayA,5,arrayB,4);**

**print(arrayD, 9, "Merge A to B: ");**

**delete [] arrayC;**

**delete [] arrayD;**

**#ifdef \_WIN32** // \_WIN32 is used by Visual C++

**#if (\_MSC\_VER <= 1916)** // check if it Visual Studio 2017 or earlier

**system("pause");**

**#endif**

**#endif**

**return 0;**

**}**

**int\* append(int\* arrayA, int sizeA, int\* arrayB, int sizeB)**

**{**

**// fill your codes here**

**}**

**int\* merge(int\* arrayA, int sizeA, int\* arrayB, int sizeB)**

**{**

**// fill your codes here**

**}**

**void print(int\* array, int size, const char \* lable)**

**{**

**// fill your codes here**

**}**

**// paste your outputs here to verify your routine is working**