Name: ________Hong Hai Dang Nguyen_________ Student ID:___103503191___

You will need:
Online C++ IDE or Computer installed IDE
A computer with internet access
A computer with internet

SWE20004 Technical Software Development

Lab 8 (week 8)

In this lab you will investigate C++ Addresses and Pointers

Before you start the lab exercise, download an appropriate C++ IDE to run your commands and programs.

1. Answer the following questions? (Look it up with Google). Don't copy and paste – write down what it is in your own words.

1.1 What is a pointer variable can store?

A pointer variable can store a memory address. In other words, in pointers, the addresses of other variables or memory elements are stored.

1.2 which operator is used to determine the address of a variable in C++?

'&' is used to determine the address of a variable in C++.

1.3 Pointer variables must always be ________ before they are used.

Pointer variables must always be created and initialised before they are used.

1.4 In order to follow a pointer and access the data that it is pointing to, we must ________ the pointer using the ________ operator.

In order to follow a pointer and access the data that it is pointing to, we must dereference the pointer using the indirection operator *.

1.5 Pointers can be used to dynamically allocate storage from the ________ at ________.

Pointers can be used to dynamically allocate storage from the heap memory at runtime.

1.6 When using raw pointers and dynamic storage allocation, we must always de-allocate the used storage by using ________ to prevent ________.

When using raw pointers and dynamic storage allocation, we must always de-allocate the used storage by using delete operator to prevent any run-time exceptions from arising.

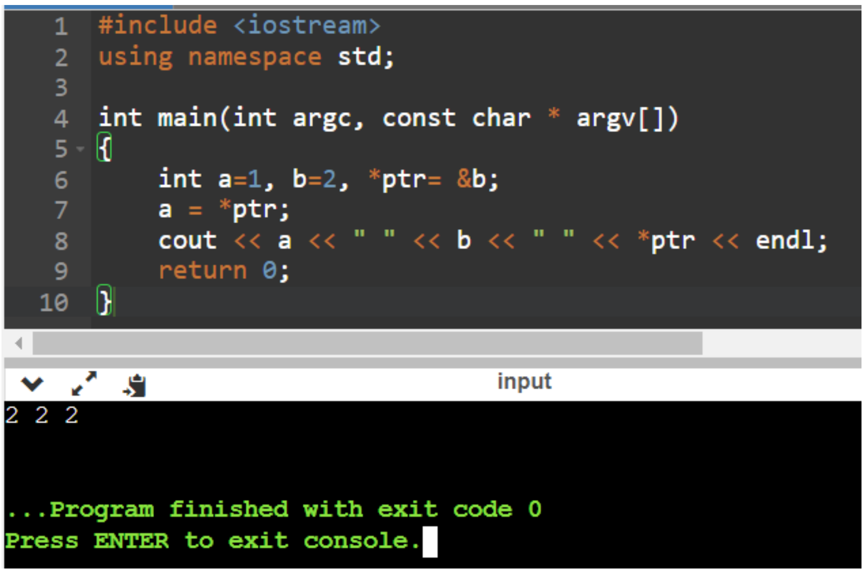
Name: ________Hong Hai Dang Nguyen_________ Student ID:___103503191___

2. Give the values of the variables after these statements are executed: (write the program and run it to find the values of the variables)

   The full program with the output will be provided below each question. The answer for all questions will be provided in a table.

a)

```
int a=1, b=2, *ptr=&b;
...
a = *ptr;
```

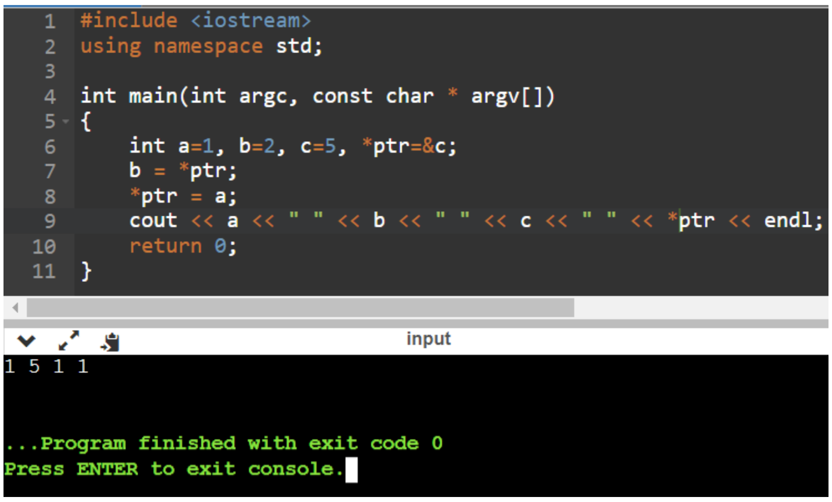


```
#include <iostream>
using namespace std;

int main(int argc, const char * argv[])
{
    int a=1, b=2, *ptr= &b;
    a = *ptr;
    cout << a << " " << b << " " << *ptr << endl;
    return 0;
}
```

```
2 2 2


...Program finished with exit code 0
Press ENTER to exit console.
```

b)

```
int a=1, b=2, c=5, *ptr=&c;
...
b = *ptr;
*ptr = a;
```



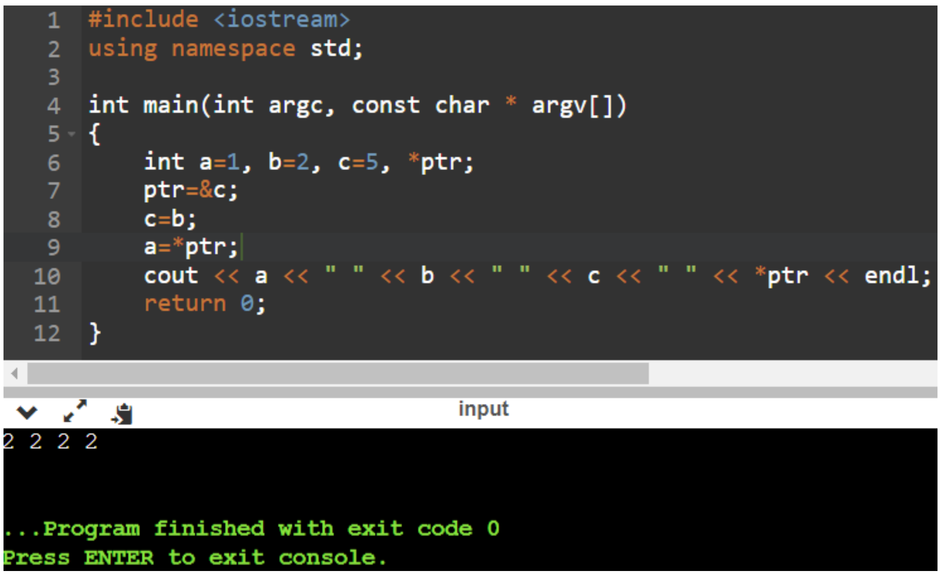
```
#include <iostream>
using namespace std;

int main(int argc, const char * argv[])
{
    int a=1, b=2, c=5, *ptr=&c;
    b = *ptr;
    *ptr = a;
    cout << a << " " << b << " " << c << " " << *ptr << endl;
    return 0;
}
```

```
1 5 1 1


...Program finished with exit code 0
Press ENTER to exit console.
```

Name: ________Hong Hai Dang Nguyen_________ Student ID:___103503191___

c)

int a=1, b=2, c=5, *ptr;
...
ptr=&c;
c=b;
a=*ptr;

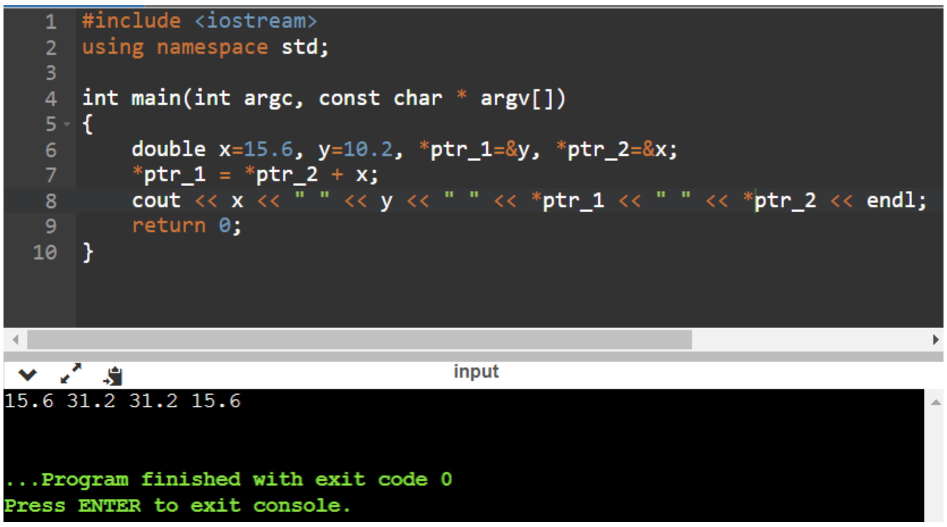


```
#include <iostream>
using namespace std;

int main(int argc, const char * argv[])
{
    int a=1, b=2, c=5, *ptr;
    ptr=&c;
    c=b;
    a=*ptr;
    cout << a << " " << b << " " << c << " " << *ptr << endl;
    return 0;
}
```

```
2 2 2 2


...Program finished with exit code 0
Press ENTER to exit console.
```

d)

double x=15.6, y=10.2, *ptr_1=&y, *ptr_2=&x;
...
*ptr_1 = *ptr_2 + x;

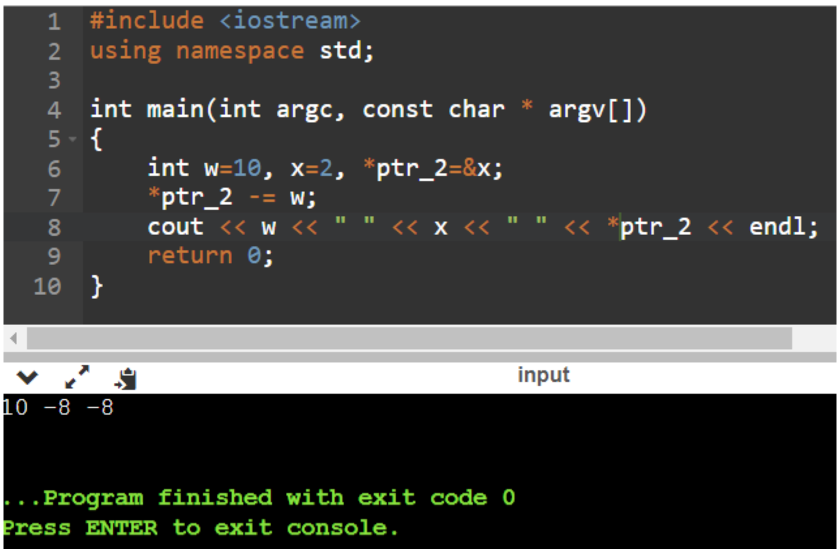


```
#include <iostream>
using namespace std;

int main(int argc, const char * argv[])
{
    double x=15.6, y=10.2, *ptr_1=&y, *ptr_2=&x;
    *ptr_1 = *ptr_2 + x;
    cout << x << " " << y << " " << *ptr_1 << " " << *ptr_2 << endl;
    return 0;
}
```

```
15.6 31.2 31.2 15.6


...Program finished with exit code 0
Press ENTER to exit console.
```

e)

int w=10, x=2, *ptr_2=&x;
...
*ptr_2 -= w;



```
#include <iostream>
using namespace std;

int main(int argc, const char * argv[])
{
    int w=10, x=2, *ptr_2=&x;
    *ptr_2 -= w;
    cout << w << " " << x << " " << *ptr_2 << endl;
    return 0;
}
```

```
10 -8 -8

...Program finished with exit code 0
Press ENTER to exit console.
```

Final answers:

a) a = 2, b = 2, *ptr = 2;

b) a = 1, b = 5, c =1, *ptr = 1

c) a = 2, b = 2, c = 2, *ptr =2

d) x = 15.6, y = 31.2, *ptr_1 = 31.2, *ptr_2 = 15.6

e) w=10, x=-8, *ptr_2 = -8

3. Show the contents of the variables and pointers that have changed after executing each of the following codes, here **m** is name of an array.

   int i = 10,*p1,*p3;

   int m[]={2,4,8,9,12,32,78,54,98};

   a) p1 = &i;

   *p1 = 8;

   b) p1 = &m[0]; p1 = p1 + 2;

   *p1 = *p1 + 8;

   c) p1 = &i; p3 = m;

   *(p3 + 1) = *p3 + *p1;

   d) p1 = m + 2; p3 = p1 + 1;

i = *p1 + *p3;

a)
p1 = &i, //p1 is pointing to i
*p1 = 8 //assigning 8 to address pointed by p1. i.e., i = 8
m and p3 remain the same.

b)
p1 = &m[0]; //p1 is pointing to first element of array m
p1 = p1 + 2; //p1 is pointing to next second element of p1.
i.e p1 is pointing to third element of array m, m[2]
*p1 = *p1 + 8; //incrementing value at address pointed by p1 by 8.
i.e, m[2] = m[2] + 8 => m[2] = 8 + 8 => m[2] = 16
*p1 = 16
The remaining elements are unchanged.

c)
p1 = &i //p1 is pointing to i
p3 = m //p3 is pointing to first element of m, by default array name points to first element of array => *p3 = 2
*(p3 + 1) = *p3 + *p1
// (p3 + 1) pointing to next first element of array p3, i.e, second element of array m => m[1]
// *p3 is value of m[0] and *p1 is value of i
*(p3 + 1) = *p3 + *p1 => m[1] = m[0] + i => m[1] = 2 + 8 => m[1] = 10
*(p3 + 1) = 10

d)
*p1 = 8, *p3 = 9; i=17

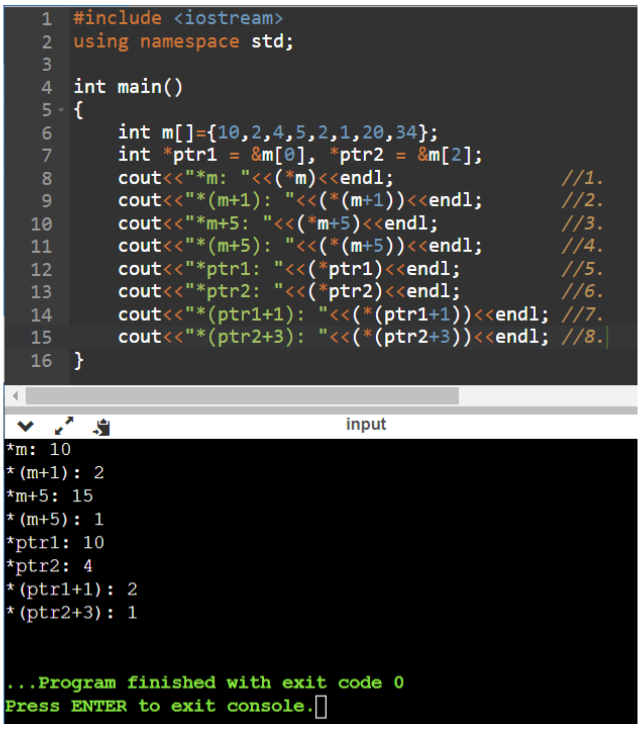
4. Assume that an array **m** is defined with the following statement:

```
int  m[]={10,2,4,5,2,1,20,34};
int*ptr1=&m[0],*ptr2=&m[2];
```

Give the value of the following references only **(The answers will be highlighted in red next to the references)**:

1. *m = 10
2. *(m+1) = 2
3. *m+5 = 15
4. *(m+5) = 1
5. *ptr1 = 10
6. *ptr2 = 4
7. *(ptr1+1) = 2
8. *(ptr2+3) = 1

Name: ________Hong Hai Dang Nguyen_________ Student ID:___103503191___



```
#include <iostream>
using namespace std;

int main()
{
    int m[]={10,2,4,5,2,1,20,34};
    int *ptr1 = &m[0], *ptr2 = &m[2];
    cout<<"*m: "<<(*m)<<endl;                    //1.
    cout<<"*(m+1): "<<(*(m+1))<<endl;            //2.
    cout<<"*m+5: "<<(*m+5)<<endl;                //3.
    cout<<"*(m+5): "<<(*(m+5))<<endl;            //4.
    cout<<"*ptr1: "<<(*ptr1)<<endl;              //5.
    cout<<"*ptr2: "<<(*ptr2)<<endl;              //6.
    cout<<"*(ptr1+1): "<<(*(ptr1+1))<<endl;      //7.
    cout<<"*(ptr2+3): "<<(*(ptr2+3))<<endl;      //8.
}
```

input

```
*m: 10
*(m+1): 2
*m+5: 15
*(m+5): 1
*ptr1: 10
*ptr2: 4
*(ptr1+1): 2
*(ptr2+3): 1


...Program finished with exit code 0
Press ENTER to exit console.
```

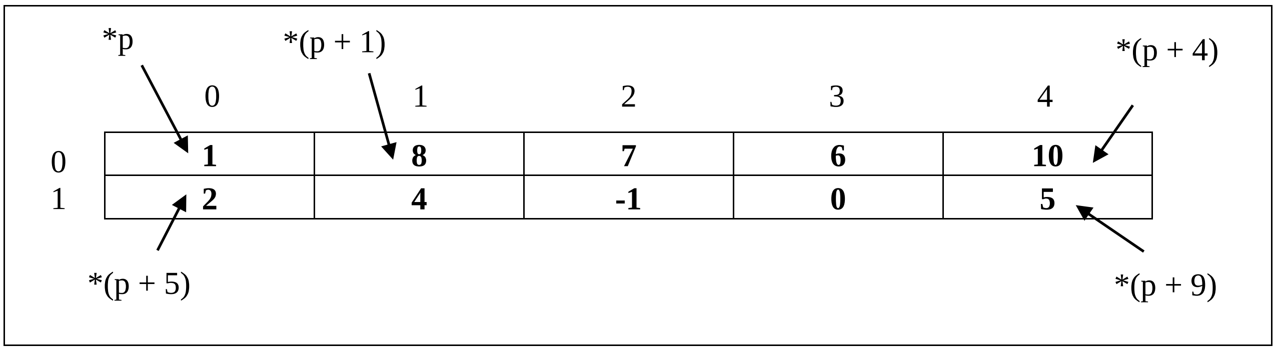
5. Assume that an integer array ***m*** is defined by the following statements:

   int m[2][5]={{1,8,7,6,10},{2,4,-1,0,5}},*p=&m[0][0];

   Draw a memory allocation **diagram**, and give the value indicated by each of the following references:

   1. *p
   2.*(p+2)
   3. *p +2
   4. *(p+1) + *(p+5)

1. *p = 1;
2. *(p+2) = 7
3. *p + 2 = 3
4. *(p+1) + *(p+5) = 10



	0	1	2	3	4
0	**1**	**8**	**7**	**6**	**10**
1	**2**	**4**	**-1**	**0**	**5**

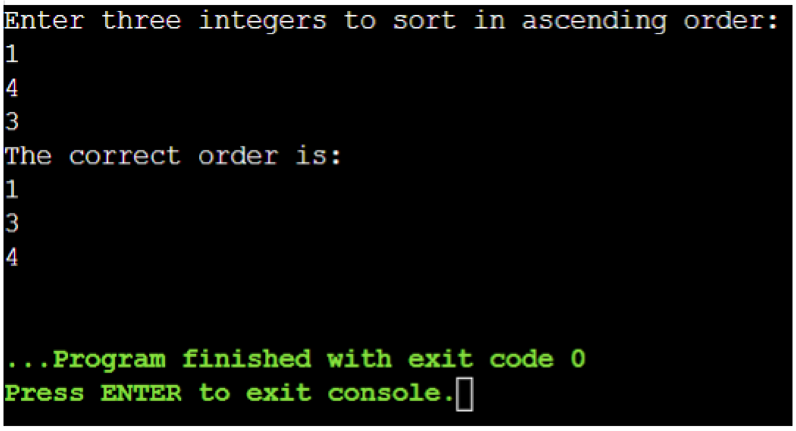
Name: ________Hong Hai Dang Nguyen_________ Student ID:___103503191___

6. Write a function that reorders the values in three integer variables such that the values are in ascending order. Assume that the following function prototype is used.

   void reorder (int *a, int *b, int*c);

   where a, b, and c are pointers to the three variables. Write a main function to test the reorderfunction.

```
#include <iostream>
using namespace std;
void reorder();
void reorder(int *a, int *b, int *c)
{
 {
//using swap function to order the integers
    if (*a > *b)
    {
        swap(*a, *b);
    }

    if (*b > *c)
    {
        swap(*b, *c);
    }

    if (*a > *b)
    {
    swap(*a, *b);
    }
 }
 cout<<"The correct order is: \n";
 cout<<*a<<"\n";
 cout<<*b<<"\n";
 cout<<*c<<"\n";
}

int main()
{
//input the three integers
 int a,b,c,_min,_mid,_max;
 cout<<"Enter three integers to sort in ascending order:"<<endl;
 cin>>a;
 cin>>b;
 cin>>c;
 reorder(&a,&b,&c);
 return 0;
}
```



```
Enter three integers to sort in ascending order:
1
4
3
The correct order is:
1
3
4


...Program finished with exit code 0
Press ENTER to exit console.
```

Report (SWE20004)

Write a one-page report on this lab covering the following:

1. Summarize the topics you explored and the activities you did during this lab.
2. Classify (group) these topics and actions under appropriate headings. Do not just copy the headings used in the instructions. For example, explain the following, what are the following commands do?
3. Discuss the relevance of these topics and actions in terms of C++ programming. i.e. How do the things in this lab work contribute to your understanding of C++ programming overall?
4. Why do you need to understand (and use) C++ Addresses and Pointers libraries?

Introduction

This week's report will illustrate the topics covered and the activities of the lab. More specifically, we were taught about C++ Addresses and Pointers libraries. In addition, we went over some of the exercises in this week's lab in order to strengthen our knowledge and understanding of the topics. This is also the final practical activity for this unit.

Addresses

A reference variable is created using the '&' operator. However, it can also be used to find a variable's memory address, or the place on the computer where the variable is kept. In C++, a memory address is given to each variable when it is created. And when we give the variable a value, that value is saved in this memory location. Use the '&' operator to access it, and the outcome will show where the variable is kept.

```
string phone = "iPhone";

cout << &phone;
```

Pointers

A variable that contains the address of another variable is known as a pointer. Before using a pointer, just like any variable or constant, you must declare it. A pointer variable declaration's general format is:

```
type *var-name;
```

Here, var-name is the name of the pointer variable, and type is the base type of the pointer; it must be a legitimate C++ type. The same asterisk that you used to declare a pointer is also used for multiplication. However, the asterisk in this sentence indicates that a variable is a pointer.

All pointers, regardless of whether they are integer, float, character, or another type, actually have the same value, which is a lengthy hexadecimal number that corresponds to a memory address. The data type of the variable or constant that the pointer points to is the only distinction between pointers of various data kinds.

There are a few crucial procedures that we will use pointers for quite frequently. A pointer variable is defined, then we give a pointer the address of a variable. Finally, use the address in the pointer variable to access the value. Using the unary operator *, which gives the value of the variable located at the address given by its operand, accomplishes this.

Lab activities

This week’s lab exercises were relatively difficult in terms of giving explanations for the answers. As for the coding, it is not as challenging as week 7’s lab, yet it still costs me a large amount of time to finish it.

The importance of this week’s topic

Once you realize the significance of references and pointers in C++, you will be able to use them to control data in the computer's memory, which will help you write less code and achieve better performance. It is also worth mentioning that one aspect that distinguishes C++ from other programming languages like Python and Java is these two features.

Conclusion

Overall, this week's lab helped us to understand the Addresses and Pointers libraries. I find myself not using these features much in my programs. Given that I have understood the importance of the two features (as mentioned above), I will try my best to implement this knowledge in my future codes to achieve better performance and efficiency in using C++.