Name: _______Hong Hai Dang Nguyen_______ Student ID:___103503191____

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

SWE20004 Technical Software Development

Lab 3 (week 3)

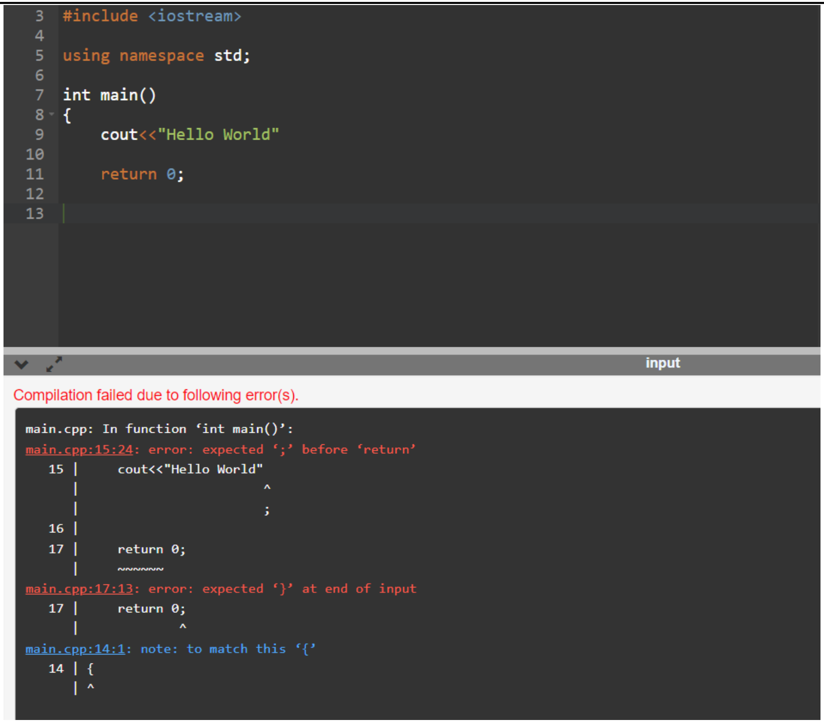
In this lab you will investigate C++ Expressions, Errors and Statements

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

1. Answer the following questions? Don't copy and paste – write down what it is - in your own words briefly.

1.1 What are compiler errors? Give an example

Compiler errors are the errors encountered as the programmer violated the standards of C++ syntax. This compiler error means that the code cannot be compiled until the errors are resolved. For example: Missing a semi-colon or missing parenthesis.



```
#include <iostream>

using namespace std;

int main()
{
    cout<<"Hello World"

    return 0;

```

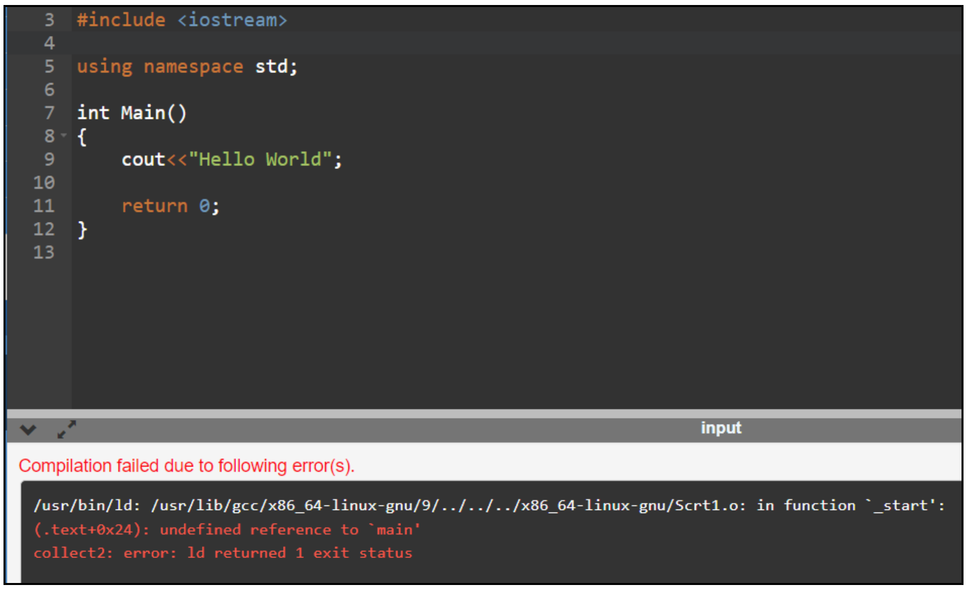
Compilation failed due to following error(s).

```
main.cpp: In function ‘int main()’:
main.cpp:15:24: error: expected ‘;’ before ‘return’
   15 |     cout<<"Hello World"
      |                        ^
      |                        ;
   16 |
   17 |     return 0;
      |     ~~~~~~
main.cpp:17:13: error: expected ‘}’ at end of input
   17 |     return 0;
      |             ^
main.cpp:14:1: note: to match this ‘{’
   14 | {
      | ^
```

1.2 What are Linker errors? Give an example

Linker errors are the errors encountered when we link multiple object files with the main's object after the compilation. These errors happen when the executable of the program cannot be created. It is possibly the result of faulty function prototypes or using the wrong header files. For example: Using Main() instead of main().

Name: _______Hong Hai Dang Nguyen_______ Student ID:___103503191____



```
#include <iostream>

using namespace std;

int Main()
{
    cout<<"Hello World";

    return 0;
}
```

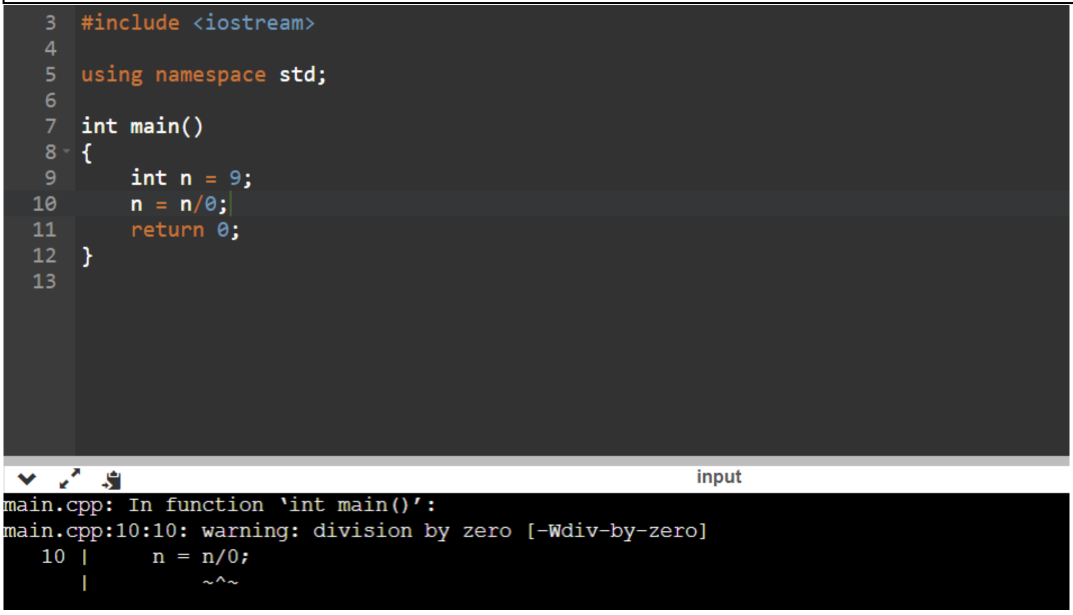
input

Compilation failed due to following error(s).

```
/usr/bin/ld: /usr/lib/gcc/x86_64-linux-gnu/9/../../../x86_64-linux-gnu/Scrt1.o: in function `_start':
(.text+0x24): undefined reference to `main'
collect2: error: ld returned 1 exit status
```

1.3 What are Runtime errors? Give an example

Runtime errors are the errors encountered during the program execution (i.e run-time) following the successful compilation. <u>For example:</u> The divison by zero.



```
#include <iostream>

using namespace std;

int main()
{
    int n = 9;
    n = n/0;
    return 0;
}
```

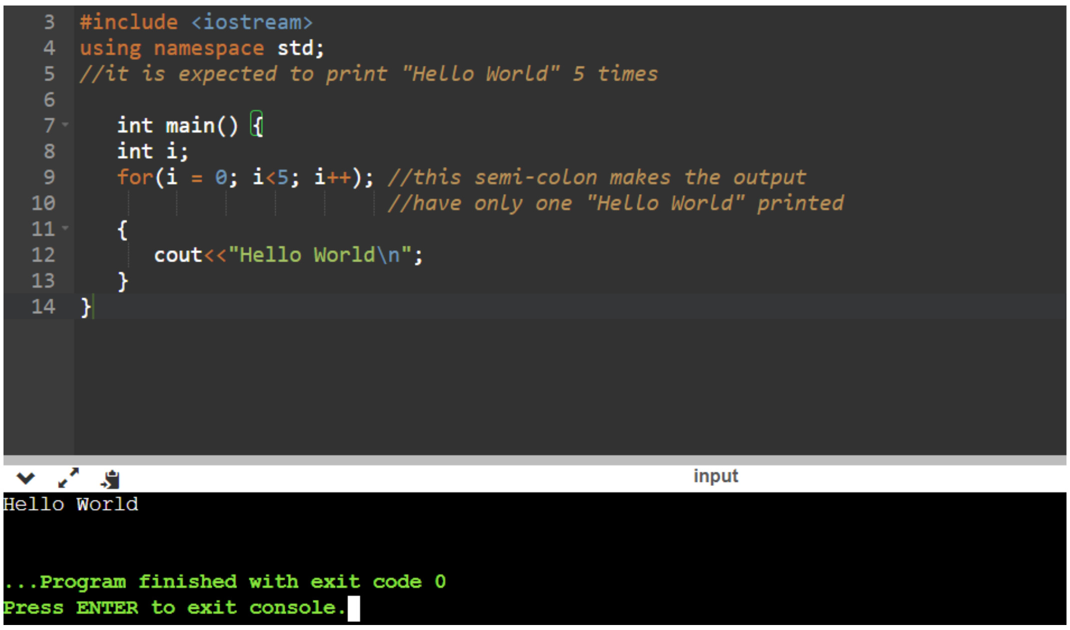
input

```
main.cpp: In function ‘int main()’:
main.cpp:10:10: warning: division by zero [-Wdiv-by-zero]
   10 |     n = n/0;
      |         ~^~
```

1.4 What are Logic errors? Give an example

Logic errors are the errors encountered when the provided inputs during the compilation and execution of the program generate indesirable results yet is error-free. This is considered to be the one of the most common programming errors for new programmers. <u>For example:</u> Using a semicolon after loop.

Name: _______Hong Hai Dang Nguyen_______ Student ID:___103503191____



```
#include <iostream>
using namespace std;
//it is expected to print "Hello World" 5 times

   int main() {
   int i;
   for(i = 0; i<5; i++); //this semi-colon makes the output
                         //have only one "Hello World" printed
   {
      cout<<"Hello World\n";
   }
}
```

```
Hello World


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

2. Rewrite the following C++ code after removing any/all syntactical errors with each correction underlined.

Note : Assume all required header files are already included in the program.

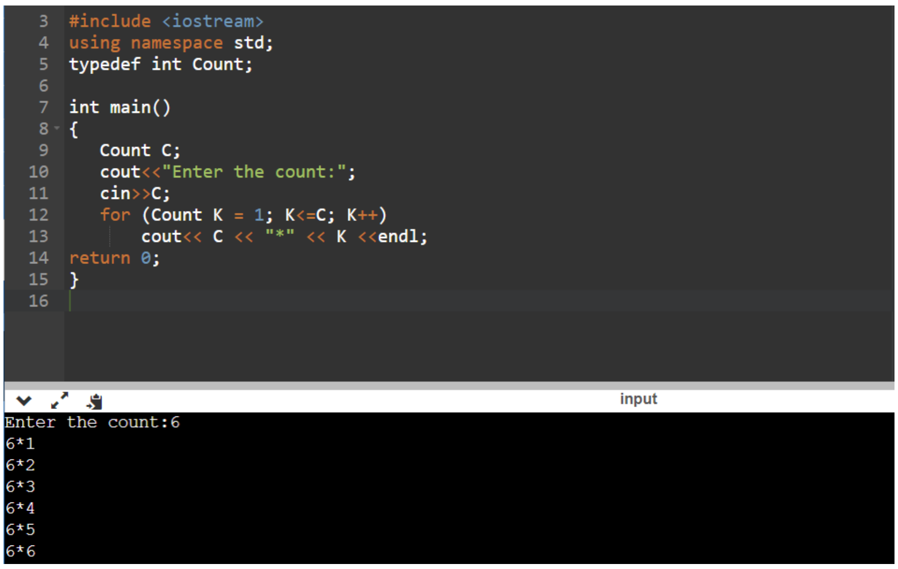
```
Typedef Count int;
void main()
{
        Count C;
        cout<<"Enter the count:";
        cin>>C; for (K = 1; K<=C; K++)
        cout<< C "*" K <<endl;
```

The errors fixed will be highlighted in <u>underlined green</u>

<u>typedef</u> <u>int Count</u>;

<u>int</u> main()
{
  Count C;
  cout<<"Enter the count:";
  cin>>C;
  for (<u>Count</u> K = 1; K<=C; K++)
     cout<< C<u> << </u>"*" <u><<</u> K <<endl;
<u>return 0;</u>
<u>}</u>

Name: _______Hong Hai Dang Nguyen_______ Student ID:___103503191____



```
#include <iostream>
using namespace std;
typedef int Count;

int main()
{
    Count C;
    cout<<"Enter the count:";
    cin>>C;
    for (Count K = 1; K<=C; K++)
        cout<< C << "*" << K <<endl;
return 0;
}
```

```
Enter the count:6
6*1
6*2
6*3
6*4
6*5
6*6
```

3. Write the function call or statement that performs these operations:

   x^y __pow(x,y)__

   23^2 ___23*23___

   i = i + 1 __i++__

   convert x degrees to y radians ___x*(3.1416/180)___

4. What function allows text with spaces (or 'whitespace') in it to be read from the keyboard into a string variable (C++ string type)?

   Using the getline() function will help us, as this allows text with spaces (or 'whitespace') in it to be read from the keyboard into a string variable.

5. What do these manipulators and flags do?

- fill(): specifies the character used to pad out extra space in a field (when width() is used). Accepts the character as an argument.

- setf(): the "set flags" function. Accepts as a parameter the flag to be enabled. Some of the flags can be activated or deactivated.

- unsetf(): the "unset flags" function. Call this to disable one of the flags.

Expression	**Explanation**
cout.fill('#')	Specify the fill character.

cout.setf(ios::flush)	Flush the output buffer to the output device before processing continues.
cout.setf(ios::right)	If this is set, output items will be right-justified inside the field (when width() or setw() is used) and empty spaces will be filled with the fill character (the space, by default).
cout.setf(ios::showpos)	If this option is enabled, positive numbers will be preceded by a plus sign. Can be reset by manipulator ***noshowpos***.
cout.setf(ios::scientific)	If this option is enabled, floating-point values are displayed in scientific notation (exponential). When this flag is set, ***ios::fixed*** is unset by default.
cout.setf(ios::left)	If this is set, output items will be left-justified inside the field (when width() or setw() is used) and empty spaces will be filled with the fill character (the space, by default).
cout.width(n)	Define the "field width" of the subsequent output item. As a parameter, the number of character places is provided. When this function is used to define field widths, left and right justify flags are used.
cout.unsetf(ios::fixed)	Disable the flag 'ios::fixed'.
cout.setf(ios::fixed)	If this option is enabled, floating-point precision are displayed using fixed-point notation. When this flag is set, ***ios::scientific*** is unset by default.

6. Problem – If Statement: Can you Drive?

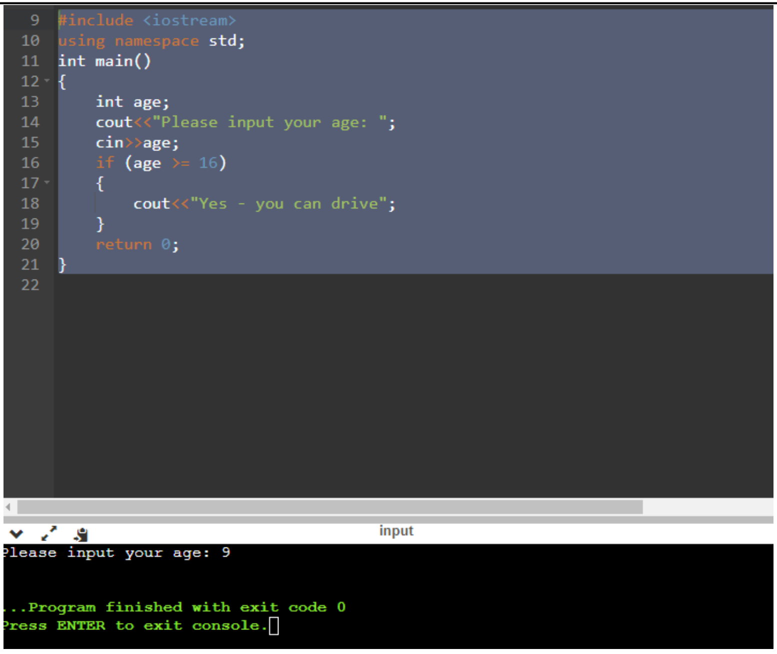
In this exercise you will use a simple if statement to decide if someone can drive.

Let's assume that anyone 16 or older can legally drive. If the person can legally drive, your program should display, "Yes- you can drive:"

If the person cannot legally drive, then your program should not display anything. The age will be provided by you. You will provide different values of age to test your code. Write and execute your code.

Name: _______Hong Hai Dang Nguyen_______ Student ID:___103503191____

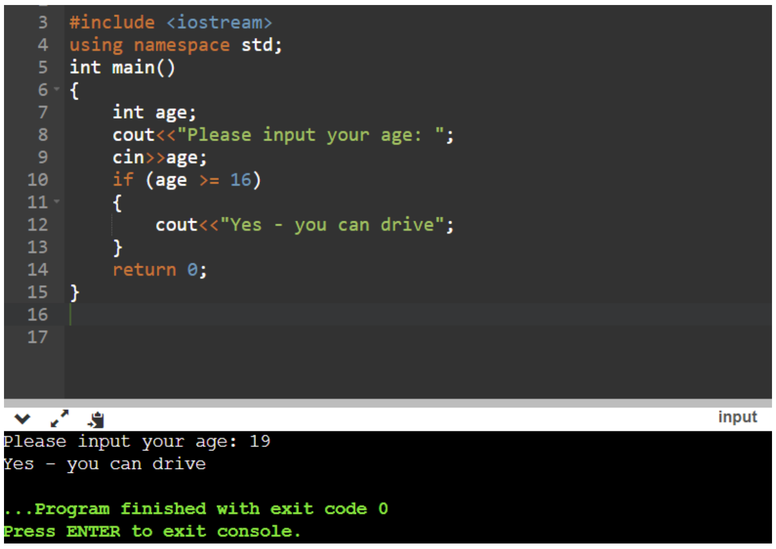
```
#include <iostream>
using namespace std;
int main()
{
   int age;
   cout<<"Please input your age: ";
   cin>>age;
   if (age >= 16)
   {
      cout<<"Yes - you can drive";
   }
   return 0;
}
```



```
#include <iostream>
using namespace std;
int main()
{
    int age;
    cout<<"Please input your age: ";
    cin>>age;
    if (age >= 16)
    {
        cout<<"Yes - you can drive";
    }
    return 0;
}
```

```
Please input your age: 9

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



```
#include <iostream>
using namespace std;
int main()
{
    int age;
    cout<<"Please input your age: ";
    cin>>age;
    if (age >= 16)
    {
        cout<<"Yes - you can drive";
    }
    return 0;
}
```

```
Please input your age: 19
Yes - you can drive

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

Name: _______Hong Hai Dang Nguyen_______ Student ID:___103503191____

7. In this exercise you will write a program that uses arithmetic operator = to change the value of an initialized variables as well as assign the value of one variable to another.

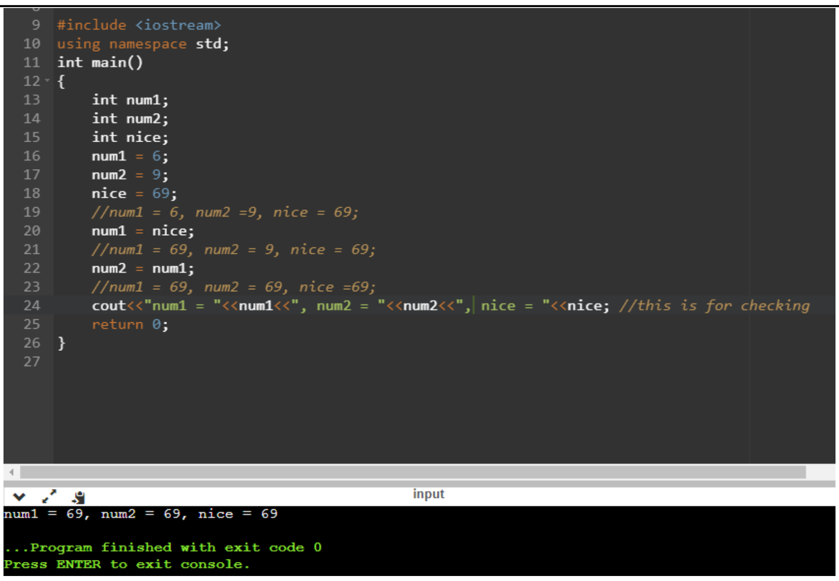
   Begin by declaring and initializing the integer variable num1 to a value of …..

   Declare and initialize the integer variable num2 to a value of …..

   Use assignment operator to change the value of num1 to …..

   Now use the assignment operator to assign the value of num1 to num2.

```
#include <iostream>
using namespace std;
int main()
{
    int num1;
    int num2;
    int nice;
    num1 = 6;
    num2 = 9;
    nice = 69;
    //num1 = 6, num2 =9, nice = 69;
    num1 = nice;
    //num1 = 69, num2 = 9, nice = 69;
    num2 = num1;
    //num1 = 69, num2 = 69, nice =69;
    cout<<"num1 = "<<num1<<", num2 = "<<num2<<", nice = "<<nice; //this is for checking
    return 0;
}
```



```
#include <iostream>
using namespace std;
int main()
{
    int num1;
    int num2;
    int nice;
    num1 = 6;
    num2 = 9;
    nice = 69;
    //num1 = 6, num2 =9, nice = 69;
    num1 = nice;
    //num1 = 69, num2 = 9, nice = 69;
    num2 = num1;
    //num1 = 69, num2 = 69, nice =69;
    cout<<"num1 = "<<num1<<", num2 = "<<num2<<", nice = "<<nice; //this is for checking
    return 0;
}
```

```
num1 = 69, num2 = 69, nice = 69

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

Name: _______Hong Hai Dang Nguyen_______ Student ID:___103503191____

8. Programming Problem

Create a C++ program that asks the user for their favorite number between 1 and 100
then read this number from the console.

Suppose the user enters 24.
Then display the following to the console:

```
Amazing!! That's my favorite number too!
No really!!, 24 is my favorite number!
```

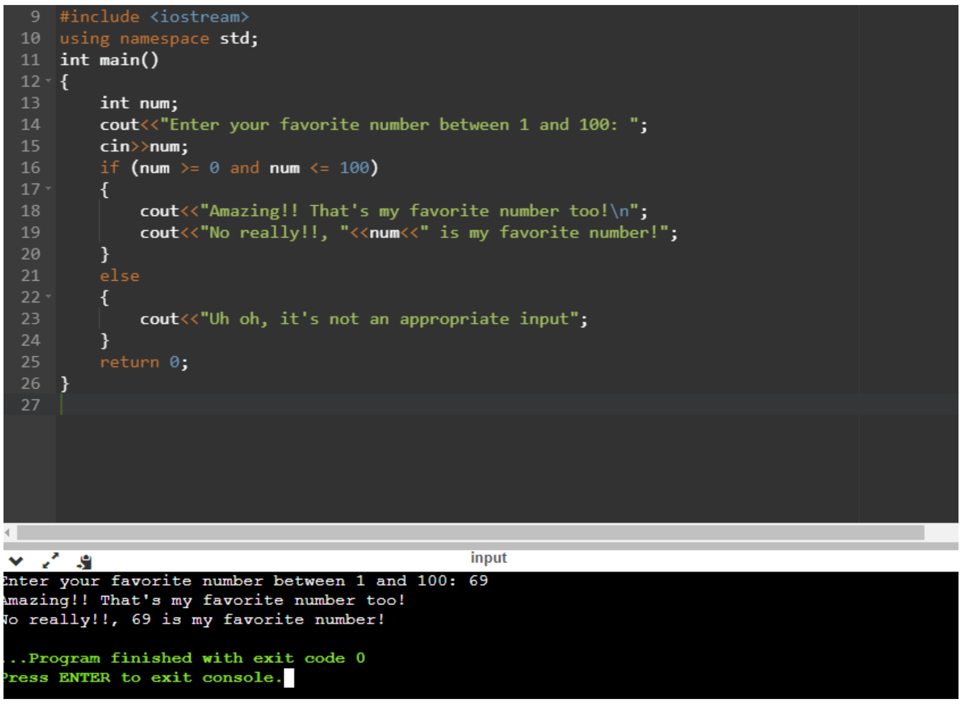
Below are 2 sample runs of the program:

```
======================================
Enter your favorite number between 1 and 100: 24
Amazing!! That's my favorite number too!
No really!!, 24 is my favorite number!

Enter your favorite number between 1 and 100: 75
Amazing!! That's my favorite number too!
No really!!, 75 is my favorite number!
```

```cpp
#include <iostream>
using namespace std;
int main()
{
    int num;
    cout<<"Enter your favorite number between 1 and 100: ";
    cin>>num;
    if (num >= 0 and num <= 100)
    {
        cout<<"Amazing!! That's my favorite number too!\n";
        cout<<"No really!!, "<<num<<" is my favorite number!";
    }
    else
    {
        cout<<"Uh oh, it's not an appropriate input";
    }
    return 0;
}
```

Name: _______Hong Hai Dang Nguyen_______ Student ID:___103503191____



```
#include <iostream>
using namespace std;
int main()
{
    int num;
    cout<<"Enter your favorite number between 1 and 100: ";
    cin>>num;
    if (num >= 0 and num <= 100)
    {
        cout<<"Amazing!! That's my favorite number too!\n";
        cout<<"No really!!, "<<num<<" is my favorite number!";
    }
    else
    {
        cout<<"Uh oh, it's not an appropriate input";
    }
    return 0;
}
```

input

```
Enter your favorite number between 1 and 100: 69
Amazing!! That's my favorite number too!
No really!!, 69 is my favorite number!

...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++ **Expressions, Errors and Statements**?

This report is worth 5% towards your unit assessment. Use the below page as a template. Either you can type it or write it in your words.

Name: _______Hong Hai Dang Nguyen_______ Student ID:___103503191____

Introduction

This week's report will illustrate the topics covered and the activities of the lab. More specifically, we were taught about the types of errors that can be encountered during the compilation or execution of the program, the expressions and statements used in the program. 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.

Type of errors

Regarding the types of errors that can be encountered during the compilation or execution of the program, there are five error types:

The first one is the compiler errors, which can be seen when a programmer violates the C++ syntax rules. This compiler error prevents code compilation until the errors are fixed. An example of this is missing a semi-colon in a line.

The second one is the linker errors, which occur when numerous object files are linked with the main object during compilation. These errors occur when the program executable cannot be produced. It might be due to defective function prototypes or improper use of header files (e.g. using Main() rather than main ()).

The third one is the runtime errors, which occur during the execution of a program after successful compilation, such as the Zero-by-zero division in the program.

The fourth one is the logic errors that can be encountered when the inputs provided during the compilation and execution of a program provide undesirable outputs while being error-free. This is one of the most frequent programming errors made by novice programmers, such as using a semicolon following a loop.

The final one is the semantic errors, which occur when the program's statements are incomprehensible to the compiler.

Expressions

C++ expressions include operators, constants, and variables structured in accordance with the language's principles. Additionally, it may contain function calls that return values. To compute a value, an expression may contain one or more operands and zero or more operators. Every expression provides a value, which is then assigned to the variable using an assignment operator. For example: ***(a + b) * c.*** There are eight types of expressions:

- Constant expressions
- Integral expressions
- Float expressions
- Pointer expressions
- Relational expressions
- Logical expressions
- Bitwise expressions
- Special assignment expressions

Statements

Statements are program components that are executed sequentially. Each function's body consists of a series of assertions. C++ has the following types of statements:

- Labeled statements
- Expression statements
- Compound statements
- Selection statements
- Iteration statements
- Jump statements
- Declaration statements
- Try blocks
- Atomic and synchronized blocks

Lab activities

This week’s lab exercises were more challenging than the previous weeks, as we have to figure out the errors in the program and resolve the issues to make the program run. We also had to implement our knowledge of the ***if*** statement and the ability to assign multiple variables to some of the exercises. These tasks took a relatively large amount of time during the lab.

The importance of this week’s topic

Having a good understanding of the types of errors will reduce the likelihood of making mistakes during the compilation of the program, and making good use of the statements and expressions in the program will reduce the time to compile a program and become more efficient and effective when programming in C++. Therefore, this week’s topics cannot be neglected.

Conclusion

Overall, this week's lab helped us to understand further the types of errors during the compilation of the program and the statements or expressions used in the program. I feel more confident in my ability to understand the C++ language and know how to create a working and effective program with fewer errors in it.