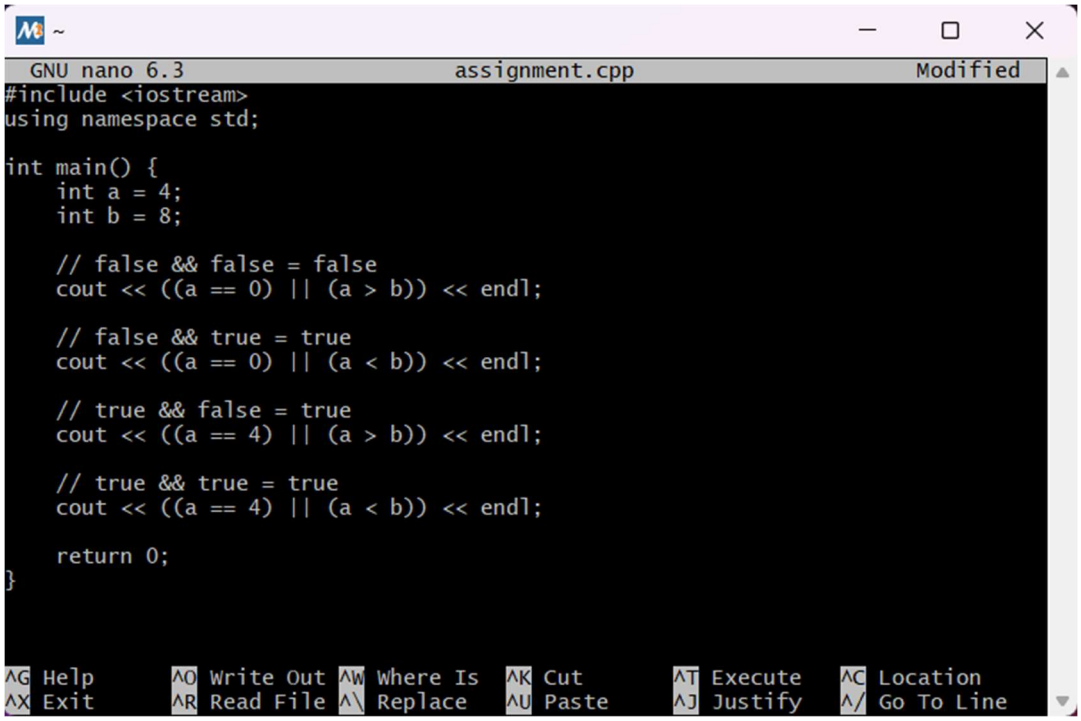
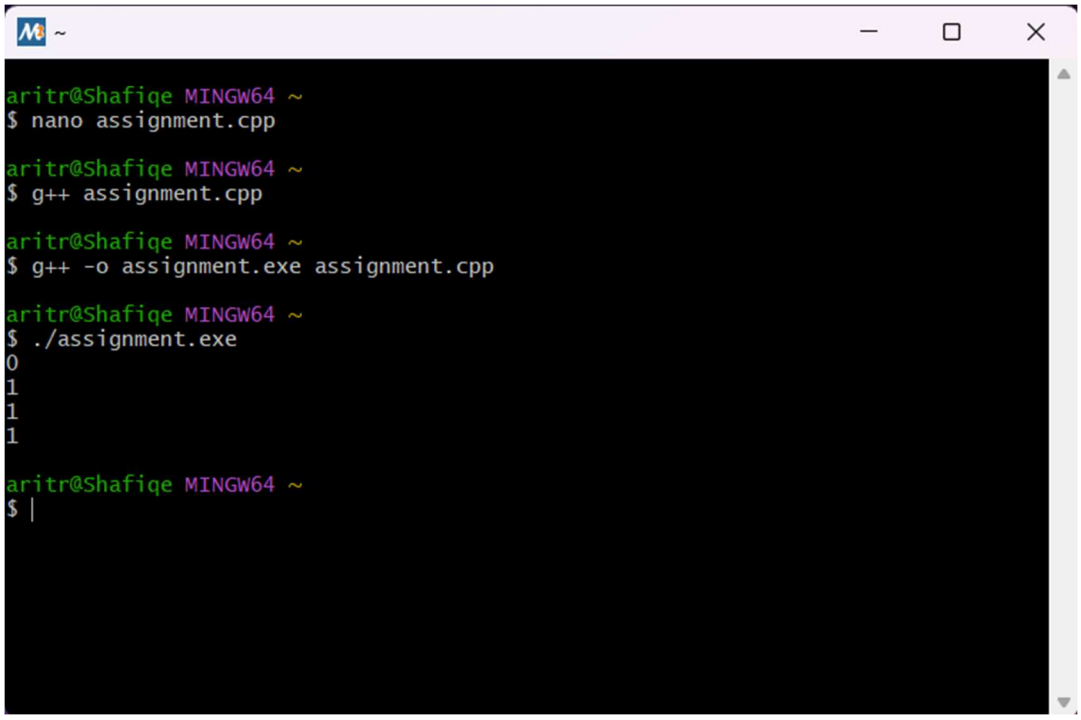
Task Description

This task asks you to write a program that uses the Clang (https://clang.llvm.org/) frontend of the Flex/Bison tools (http://www.jonathanbeard.io/tutorials/FlexBisonC++) to transform an arbitrary piece of C++ code to a new one with the relational and logical operators at IF statements reversed.

Example of C++ Logical OR Operator

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

int main() {
   int a = 4;
   int b = 8;

   // false && false = false
   cout << ((a == 0) || (a > b)) << endl;

   // false && true = true
   cout << ((a == 0) || (a < b)) << endl;

   // true && false = true
   cout << ((a == 4) || (a > b)) << endl;

   // true && true = true
   cout << ((a == 4) || (a < b)) << endl;

   return 0;
}
```


~
GNU nano 6.3 assignment.cpp Modified
#include <iostream>
using namespace std;
int main() {
int a = 4;
int b = 8;
// false && false = false
cout << ((a == 0) || (a > b)) << endl;
// false && true = true
cout << ((a == 0) || (a < b)) << endl;
// true && false = true
cout << ((a == 4) || (a > b)) << endl;
// true && true = true
cout << ((a == 4) || (a < b)) << endl;
return 0;
}
^G Help ^O Write Out ^W Where Is ^K Cut ^T Execute ^C Location
^X Exit ^R Read File ^\ Replace ^U Paste ^J Justify ^/ Go To Line


~
aritr@Shafiqe MINGW64 ~
$ nano assignment.cpp
aritr@Shafiqe MINGW64 ~
$ g++ assignment.cpp
aritr@Shafiqe MINGW64 ~
$ g++ -o assignment.exe assignment.cpp
aritr@Shafiqe MINGW64 ~
$ ./assignment.exe
0
1
1
1
aritr@Shafiqe MINGW64 ~
$

Output

0
1
1
1

Explanation

In this program, we declare and initialize two `int` variables `a` and `b` with the values `4` and `8` respectively. We then print a logical expression

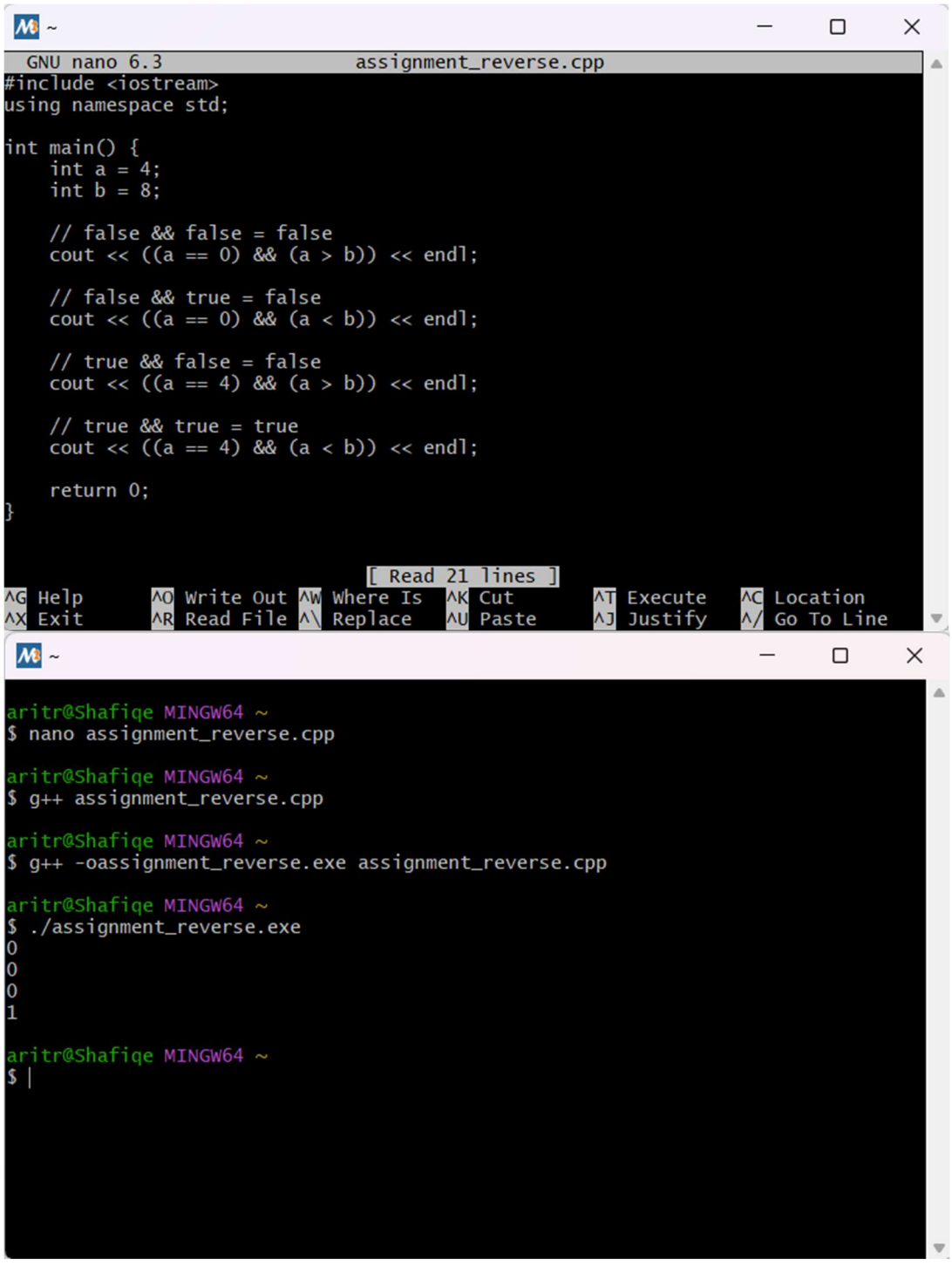
```
((a == 0) || (a > b))
```

Here, `a == 0` evaluates to `false` as the value of `a` is `4`. `a > b` is also `false` since the value of `a` is less than that of `b`. We then use the OR operator `||` to combine these two expressions.

From the truth table of `||` operator, we know that `false || false` (i.e., `0 || 0`) results in an evaluation of `false` (`0`). This is the result we get in the output.

Example of C++ OR Operator

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

int main() {
   int a = 4;
   int b = 8;

   // false && false = false
   cout << ((a == 0) && (a > b)) << endl;

   // false && true = false
   cout << ((a == 0) && (a < b)) << endl;

   // true && false = false
   cout << ((a == 4) && (a > b)) << endl;

   // true && true = true
   cout << ((a == 4) && (a < b)) << endl;

   return 0;
}
```


~
GNU nano 6.3 assignment_reverse.cpp
#include <iostream>
using namespace std;
int main() {
int a = 4;
int b = 8;
// false && false = false
cout << ((a == 0) && (a > b)) << endl;
// false && true = false
cout << ((a == 0) && (a < b)) << endl;
// true && false = false
cout << ((a == 4) && (a > b)) << endl;
// true && true = true
cout << ((a == 4) && (a < b)) << endl;
return 0;
}
[Read 21 lines]
^G Help ^O Write Out ^W Where Is ^K Cut ^T Execute ^C Location
^X Exit ^R Read File ^\ Replace ^U Paste ^J Justify ^/ Go To Line
~
aritr@Shafiqe MINGW64 ~
$ nano assignment_reverse.cpp
aritr@Shafiqe MINGW64 ~
$ g++ assignment_reverse.cpp
aritr@Shafiqe MINGW64 ~
$ g++ -oassignment_reverse.exe assignment_reverse.cpp
aritr@Shafiqe MINGW64 ~
$ ./assignment_reverse.exe
0
0
0
1
aritr@Shafiqe MINGW64 ~
$

Output

0
0
0
1

Explanation

In this program, we declare and initialize two `int` variables `a` and `b` with the values `4` and `8` respectively. We then print a logical expression

```
((a == 0) && (a > b))
```

Here, `a == 0` evaluates to `false` as the value of `a` is `4`. `a > b` is also `false` since the value of `a` is less than that of `b`. We then use the AND operator `&&` to combine these two expressions.

From the truth table of `&&` operator, we know that `false && false` (i.e., `0 && 0`) results in an evaluation of `false` (`0`). This is the result we get in the output.