Translating a C++ Program to Assembly Language

Translate C++ programs to assembly language as illustrated by the example Chapter 7/problem 57. Start with the C++ file for this problem (`p757.cpp`) in the H1 Software Package. Use the `iss` program in the software package (see `iss.txt`) to create a “`.mas`” file from the C++ file by performing the following steps:

1. Enter on the command line

   `iss p757.cpp /4` (Adds to `p757.cpp` 4 empty //-comments after each C++ statement.)

2. Using a text editor, add assembler code after each C++ statement in the empty //-comments in `p757.cpp`. Be sure to format your assembler code appropriately (left justify labels and indent a *consistent* number of columns—for example, 10—before entering a mnemonic or a directive).

3. Enter on the command line

   `iss p757.cpp /-4` (Removes unused empty //-comments from `p757.cpp`)

4. Enter on the command line

   `mas p757.cpp` (Assembles the assembler code *that appears as comments* in `p757.cpp`. Creates `p757.mac` and `p757.mas`. In `p757.mas`, the *C++ code now appears as comments*).

The programs below illustrate the transformation that occurs if the sequence above is applied to the program in `p757.cpp`.

```
#include <iostream>
using namespace std;
int x = 3;
int main()
{
   int y;
   y = x;
   cout << y << endl;
}
```

original program in `p757.cpp`

After step 1 (`iss p757.cpp /4`). `iss` adds empty comment lines.

```
//; iss of p757.cpp  Your name:
//
//
//
//
#include <iostream>
using namespace std;
int x = 3;
//
//
//
//
int main()
//
//
//
//
{
   int y;
//
//
//
//
   y = x;
//
//
//
//
   cout << y << endl;
//
//
//
//
}
//
//
//
//
```

`iss` adds four lines of empty comments

After step 2 (after adding assembler code for each C++ statement in `p757.cpp`)

```
//; iss of p757.cpp  Your name: Anthony J. Dos Reis
//
//
//
//
#include <iostream>
using namespace std;
int x = 3;
//x:        dw   3
//
//
//
int main()
//main:
//
//
//
{
   int y;
//          aloc 1
//
//
//
   y = x;
//          ld   x
//          str  0
//
//
   cout << y << endl;
//          ldr  0
//          dout
//          ldc  '\n'
//          aout
}
//          dloc 1
//          ldc  0
//          halt
//          ep   main
```

add your name here

assembler code added as comments

assembler code added as comments after each C++ statement
note consistent indentation of assembler code

put any `ep`, `public`, and `extern` directives at the end

After step 3 (`iss p757.cpp /-4`). `iss` removes unused empty comment lines

```
//; iss of p757.cpp  Your name: Anthony J. Dos Reis
#include <iostream>
using namespace std;
int x = 3;
//x:        dw   3
int main()
//main:
{
   int y;
//          aloc 1
   y = x;
//          ld   x
//          str  0
   cout << y << endl;
//          ldr  0
//          dout
//          ldc  '\n'
//          aout
}
//          dloc 1
//          ldc  0
//          halt
//          ep   main
```

After step 4 (`mas p757.cpp`). `mas` creates `p757.mas`

```
; iss of p757.cpp  Your name: Anthony J. Dos Reis
                        ; #include <iostream>
                        ; using namespace std;
                        ; int x = 3;
x:        dw   3
                        ; int main()
main:
                        ; {
                        ;    int y;
          aloc 1
                        ;    y = x;
          ld   x
          str  0
                        ;    cout << y << endl;
          ldr  0
          dout
          ldc  '\n'
          aout
                        ; }
          dloc 1
          ldc  0
          halt
          ep   main
```

`p757.mas`

To test the program in `p757.mac`:

1. Enter on the command line

   `sim p757.mac` (Runs the machine code in `p757.mac`.)

2. Enter when in the debugger

   `v` (Tests program. `sim`’s verification feature will not evaluate a program program as correct unless it is both correct and its output is *identical* to the output produced by the corresponding C++ program).

3. If debugging is required, debug the file `p757.mas`.