21.1 zyLab training: Output format errors in Pluto.cpp

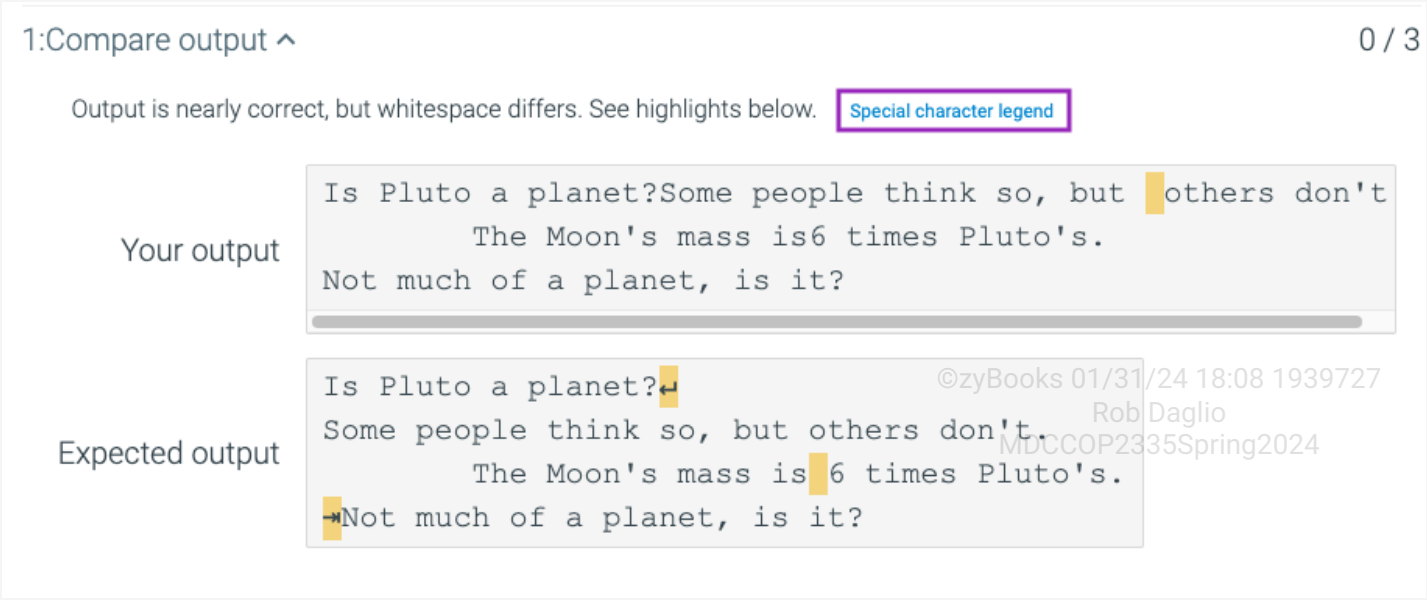
This section has been set as optional by your instructor.

In zyLabs, program correctness is not limited to producing program output containing the correct content. Program correctness also includes producing program output in the correct format. A program introducing an extra space or a missing newline character in an output is considered as not being precise.

The program in the default template is expected to produce the following output:

```
Is Pluto a planet?
Some people think so, but others don't.
    The Moon's mass is 6 times Pluto's.
    Not much of a planet, is it?
```

1. Click the **Run program** button in **Develop mode** to run the program and examine the output. The program produces an output containing the correct words as expected but in a different format.
2. Switch to **Submit mode** and click the **Submit for grading** button. The submitted program has failed the compare output test. In the image below, the auto-grader highlights the differences between the submitted program's output and the expected program's output.



Any highlighted characters in the **Your output** box represent the extraneous characters the submitted program has introduced. Any highlighted characters in the **Expected output** box represent missing

characters that the submitted program has failed to output. Click the **Special character legend** link (indicated in the compare output test result image above) to understand the meaning of different highlighted characters.

3. In the **Your output** box, a space ( ) is highlighted in between "but" and "others". Locate the output statement containing this error in the program and remove the extra space. Click **Submit for grading** and notice that the highlighted space is removed from the result of the compare output test.
4. In the **Expected output** box, a newline (↵) is highlighted after "planet?". Locate the output statement containing this error in the program and add the missing newline. Submit the program again and notice that the highlighted newline is removed from the result of the compare output test.
5. Repeat the process to add the two remaining missing characters (a space and a tab character) in the program output. Submit the program again and notice that all highlighted characters are now removed from the result of the compare output test. A total score of 3/3 is awarded for a correct compare output test.

539740.3879454.qx3zqy7

LAB ACTIVITY | 21.1.1: zyLab training: Output format errors in Pluto.cpp | 0 / 5

Pluto.cpp — Load default template...

```
1  #include <iostream>
2  using namespace std;
3
4  int main() {
5     int proportion = 6;
6
7     cout << "Is Pluto a planet?";
8     cout << "Some people think so, but  others don't." << endl;
9     cout << "\tThe Moon's mass is" << proportion << " times Pluto's." << endl;
10    cout << "Not much of a planet, is it?" << endl;
11
12    return 0;
13 }
14
```


Develop mode | **Submit mode**

Run your program as often as you'd like, before submitting for grading. Below, type any needed input values in the first box, then click **Run program** and observe the program's output in the second box.

Enter program input (optional)

If your code requires input values, provide them here.

Run program

Input (from above) → Pluto.cpp (Your program) → Outp

Program output displayed here

Coding trail of your work What is this?

```
History of your effort will appear here once you begin working on this zyLab.
```

21.2 zyLab training: Logic errors in Kite.cpp

This section has been set as optional by your instructor.

Logic errors can be subtle and difficult to detect. Whereas syntax errors will have the compiler generate error messages to give hints as to what and/or where an issue may be, logic errors may not cause the program to generate any errors at all. Programs with logic errors may compile and run, but the resultant output may be unexpected or incorrect.

This program calculates the area of a rectangular kite. However, the mathematical formula for the area is incorrect. Fix this logic error so the area of a rectangular kite is properly calculated.

The correct output of the program is:

```
Sides: 12 10
Perimeter: 44
```


539740.3879454.qx3zqy7

LAB ACTIVITY | 21.2.1: zyLab training: Logic errors in Kite.cpp 0 / 5

Kite.cpp Load default template...

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

int main() {
   int longSide = 12;
   int shortSide = 10;
   int perimeter;

   // Area formula
   perimeter = 2 * longSide + shortSide;

   cout << "Sides: " << longSide << " " << shortSide << endl;
   cout << "Perimeter: " << perimeter << endl;

   return 0;
}
```


Develop mode | Submit mode

Run your program as often as you'd like, before submitting for grading. Below, type any needed input values in the first box, then click **Run program** and observe the program's output in the second box.

Enter program input (optional)

If your code requires input values, provide them here.

Run program

Input (from above) → **Kite.cpp** (Your program) → Outp

Program output displayed here

Coding trail of your work What is this?

History of your effort will appear here once you begin working on this zyLab.

21.3 zyLab training: Syntax errors in Baseball.cpp

This section has been set as optional by your instructor.

The goal of this lab is to help students identify some of the syntax errors commonly made while writing a program. Syntax errors are detected by the compiler during compilation, and error messages are generated to describe the errors encountered.

The program in the default template is expected to produce the following output when the input is 15:

```
The Dodgers scored 15
```

However, syntax errors presented in the code prevent the program from being compiled.

1. Click the **Run program** button in **Develop mode** to run the program and examine the output. Compiler errors will be displayed instead of the program output:

```
Baseball.cpp: In function ‘int main()’:
Baseball.cpp:7:4: error: expected initializer before ‘cin’
    7 |    cin << dodgers;
      |    ^~~
Baseball.cpp:9:37: error: ‘dodgers’ was not declared in this scope
    9 |    cout << "The Dodgers scored " << dodgers << endl
      |                                     ^~~~~~~
```

Error messages are often long and technical. Do not expect the messages to make much sense when starting to learn a programming language. Use the messages as hints to locate the portion of the program that causes an error. An error message usually begins with the location where an error is detected in the program.

Ex: `Baseball.cpp: In function 'int main()':` indicates that errors are detected in the main() function in Baseball.cpp.

The second line of the error message indicates that an error appears in line seven of Baseball.cpp (`Baseball.cpp:7:4:`). The compiler expects an initializer before `cin`. The reason for this error is that the compiler treats `cin` as part of the statement in the previous line. The statement in the previous line (line five) is not terminated properly; in other words, the previous line is missing a semi-colon (;).

2. Correct the error in line five and run the program again in **Develop mode**. Notice that an even longer list of errors is displayed; however, the error about the initializer before `cin` in line seven is removed. Therefore, the fix applied in this step is correct.

More errors are detected in this step because the compiler treats line seven as an independent statement, and one error often causes additional errors further along in the program. Notice that the new errors are all located in line seven.

To proceed with the troubleshooting, focus on the first error displayed:

```
Baseball.cpp:7:8: error: no match for ‘operator<<’ (operand types are
‘std::istream’ {aka ‘std::basic_istream<char>’} and ‘int’)
    7 |    cin << dodgers;
      |    ~~~ ^~ ~~~~~~~
      |    |      |
      |    |      int
      |    std::istream {aka std::basic_istream<char>}
```


The error message indicates that the << operator is used incorrectly. The << operator is used in output statements rather than input statements. Another operator should be used in line seven.

3. Correct the error in line seven and run the program again. A new error is detected in main():

```
Baseball.cpp:9:52: error: expected ‘;’ before ‘return’
    9 |    cout << "The Dodgers scored " << dodgers << endl
      |                                                    ^
      |                                                    ;
   10 |
   11 |    return 0;
      |    ~~~~~~
```

The error message indicates that a semi-colon is missing in line nine, before the `return` in line eleven.This error message about missing a semi-colon is more precise compared to the first error message shown in the beginning of the lab.

4. Correct the error in line nine and run the program again. The program compiles successfully and produces the following output:

```
The Dodgers scored 15
```

5. Switch to **Submit mode** and click the **Submit for grading** button. The submitted program has passed all the tests and a total score of 2/2 is awarded.

LAB ACTIVITY | 21.3.1: zyLab training: Syntax errors in Baseball.cpp | 0 / 5

Baseball.cpp — Load default template...

```
1 #include <iostream>
2 using namespace std;
3
4 int main() {
5    int dodgers
```

```
5   int dodgers
6
7   cin << dodgers;
8
9   cout << "The Dodgers scored " << dodgers << endl
10
11  return 0;
12 }
```


Develop mode | Submit mode

Run your program as often as you'd like, before submitting for grading. Below, type any needed input values in the first box, then click **Run program** and observe the program's output in the second box.

Enter program input (optional)

If your code requires input values, provide them here.

Run program

Input (from above) → **Baseball.cpp** (Your program) → Outp

Program output displayed here

Coding trail of your work What is this?

History of your effort will appear here once you begin working on this zyLab.

21.4 LAB*: Program: ASCII cat

This section has been set as optional by your instructor.

Program Specifications Pictures made from keyboard characters are known as ASCII art and can be quite intricate. ASCII Art Archive provides examples. Write a program that reads a cat name (string)

and outputs the drawing using cout.

Note: a backslash \ in a string acts as an escape character, such as with a newline `\n`. So, to print a single backslash, include two backslashes next to each other. Ex: The following prints a single backslash: `cout << "\\";`

Note: this program is designed for *incremental development*. Complete each step and submit for grading before starting the next step. Only a portion of tests pass after each step to confirm progress.

Ex: If the input is:

```
Mittens
```

the output is:

```
      /\_/\
 /\  / o o \
//\\ \~(*)~/
`  \/   ^ /
   | \|| ||
   \ '|| ||
    \)()-())
My cat Mittens
```

Step 1 (2 pts). Write output statements for the first two lines of the figure. Hint: The first line starts with 6 blank spaces before the first slash symbol. Submit for grading to confirm 1 test passes.

Step 2 (2 pts). Write two more output statements for the next two lines. Hint: The third line starts a slash symbol. Submit for grading to confirm 2 tests pass.

Step 3 (2 pts). Write three more output statements to complete the figure without the text label. Submit for grading to confirm 3 tests pass.

Step 4 (4 pts). Read from input the cat name (string). Output the cat name in the format shown. Submit for grading to confirm all tests pass.

LAB ACTIVITY | 21.4.1: LAB*: Program: ASCII cat | 0 / 10

main.cpp — Load default template...

```
1 #include <iostream>
2 using namespace std;
3
```

```
4  int main() {
5
6     /* Type your code here. */
7
8     return 0;
9  }
10
```


Develop mode | Submit mode

Run your program as often as you'd like, before submitting for grading. Below, type any needed input values in the first box, then click **Run program** and observe the program's output in the second box.

Enter program input (optional)

If your code requires input values, provide them here.

Run program

Input (from above) → **main.cpp** (Your program) → Outp

Program output displayed here

Coding trail of your work What is this?

History of your effort will appear here once you begin working on this zyLab.

21.5 LAB*: Program: Phone directory

This section has been set as optional by your instructor.

Program Specifications Write a program to input a phone number and output a phone directory with five international numbers. Phone numbers are divided into four parts: 1) country code, 2) area code, 3) prefix, and 4) line number. For example, a phone number in the United States is +1 (555) 123-4567.

Note: this program is designed for *incremental development*. Complete each step and submit for grading before starting the next step. Only a portion of tests pass after each step but confirm progress.

Step 1 (2 pts). Read from input an area code, prefix, and line number (integers). Output the directory heading (two lines). Insert two blank spaces between Country and Phone and the horizontal line is created with dashes (not underscores). Output a phone number for the United States with country code +1 using proper format. Submit for grading to confirm 2 tests pass.
Ex: If the input is:

```
555  457  2345
```

The output is:

```
Country  Phone Number
-------  ------------
U.S.     +1 (555)457-2345
```

Step 2 (2 pts). Output a phone number for Brazil with country code +55 and add 100 to the prefix. The prefix variable should not change. Instead, add 100 to the prefix within the print statement. For example, `cout << ")" << (prefixNum + 100) << "-";` Submit for grading to confirm 3 tests pass.
Ex: If the input is:

```
555  457  2345
```

The output is:

```
Country  Phone Number
-------  ------------
U.S.     +1 (555)457-2345
Brazil   +55 (555)557-2345
```


Step 3 (2 pts). Output a phone number for Croatia with country code +385 and add 50 to the line number. Output a phone number for Egypt with country code +20 and add 30 to the area code. The variables should not change. Instead, add values within the print statement as in Step 2. Submit for grading to confirm 4 tests pass.
Ex: If the input is:

```
555  929  6453
```

The output is:

```
Country  Phone Number
-------  ------------
U.S.     +1 (555)929-6453
Brazil   +55 (555)1029-6453
Croatia  +385 (555)929-6503
Egypt    +20 (585)929-6453
```


Step 4 (2 pts). Output a phone number for France with country code +33 and swap the area code with the prefix. Submit for grading to confirm all tests pass.
Ex: If the input is:

```
555  929  6453
```

The output is:

```
Country  Phone Number
-------  ------------
U.S.     +1 (555)929-6453
Brazil   +55 (555)1029-6453
Croatia  +385 (555)929-6503
Egypt    +20 (585)929-6453
France   +33 (929)555-6453
```


LAB ACTIVITY | 21.5.1: LAB*: Program: Phone directory | 0 / 10

main.cpp — Load default template...

```
1  #include <iostream>
2  using namespace std;
3
4  int main() {
5
6     /* Type your code here. */
7
8     return 0;
9  }
10
```

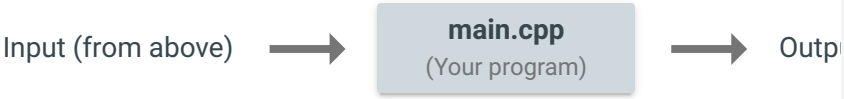
Develop mode | Submit mode

Run your program as often as you'd like, before submitting for grading. Below, type any needed input values in the first box, then click **Run program** and observe the program's output in the second box.

Enter program input (optional)

If your code requires input values, provide them here.

Run program



Program output displayed here

Coding trail of your work What is this?

```
History of your effort will appear here once you begin working
on this zyLab.
```