CS 31 Solutions Week 2

This worksheet is entirely optional, and meant for extra practice. Some problems will be more challenging than others and are designed to have you apply your knowledge beyond the examples presented in lecture, discussion or projects. All exams will be done on paper, so it is in your best interest to practice these problems by hand and not rely on a compiler.

Solutions are written in red. The solutions for programming problems are not absolute, it is okay if your code looks different; this is just one way to solve the specific problem.

Concepts
If Statements, Cin, Variables, Doubles, Ints

Reading Problems

1. Assume that the following lines of code are inside the main function, with #include <iostream> and using namespace std, and all the string variables used have been previously declared.
   (a) Circle where the bug occurs.
   (b) Explain what you think will happen when running the program.
   (c) Is this a logic error or a compilation error? Why?
   (d) Add a fix to the problem you found in part (a).

```cpp
string name;
cout << "Enter your name: ";
getline(cin, name);

cout << "\nEnter your UID: ";
int UID;
cin >> UID;
cin.ignore(10000, '
'); // add this

cout << "\nEnter your Major: ";
getline(cin, major);

cout << "\nEnter your residence hall: ";
getline(cin, hall);

cout << "\n" << UID << " is the ID of " << name << ", a " << major << " student who lives in " << hall << endl;
```
a) Bug: It will skip “Enter your Major”, because getline has already consumed a newline character.

b) A newline is always appended to your input when you select Enter or Return when submitting from a terminal. It is also used in files for moving toward the next line. When the flow of control reaches std::getline(), the newline will be discarded, but the input will cease immediately. The reason this happens is because the default functionality of this function dictates that it should (it attempts to read a line and stops when it finds a newline).

c) This is more of a logic error than a compilation error because it will not stop your program from compiling. However, because we have made a mistake in predicting what the computer will do, we have made an error in our logic. Errors like this can sneak up on you without the compiler letting you know.

d) Because this leading newline inhibits the expected functionality of your program, it follows that it must be skipped or ignored somehow. One option is to call cin.ignore(10000, '\n') after the first extraction. It will discard characters up to the next newline so that the newline is no longer intrusive.

2. What is the output of the following code?

```cpp
int cookies = 12;
int mms = 120;
if (mms % cookies != 0) {
    cout << "Can’t evenly split M&Ms for each cookie!" << endl;
} else {
    cout << "We have " << mms/cookies << " M&Ms per cookie." << endl;
}
```

“We have 10 M&Ms per cookie.”

Since the condition in the if statement isn’t met, the else statement is executed.

3. This code snippet takes a certain “hour” and “weekday” and tries to tell you if you can buy turnips from Daisy Mae, the turnip seller.
   (a) Find the 5 lines with mistakes in the code and fix them.
   (b) Will this code compile? Why or why not?
(c) After you fix the bugs, imagine you input 11 for the hour, then “Monday” for the weekday. What will this program say to you?

```cpp
int hour;
string weekday;
cin >> hour;
cin >> weekday;

if (weekday != "Sunday" || hour >= 12) {
    cout >> "Daisy Mae is not here!" >> endl;
} else {
    if (hour = 11) {
        cout >> "It's almost 12! Hurry up!" >> endl;
    } else {
        cout >> "Buy turnips with Bells." >> endl;
    } (was not in the program originally!)
}
```

a) The arrows for the output statements are in the wrong direction (should be << instead of >>). There should also be a double equal (==) in the if statement.

b) This code will not compile! The line with the conditional (hour = 11) will not be a compile error but would be a logic error because there will be a variable assignment instead of an expected truth value. All of the other errors will cause compilation errors. It's important to be very careful as you read and write code on paper. There won't be any compiler to warn you during the exam that you switched the symbols for cout and cin, or that you forgot one bracket or equals symbol! :(

c) “Daisy Mae is not here!”

4. What will this program output? Can you explain every line of output?

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

int main() {
    int elligent = 64;
    int eresting = 0;
    double rainbow = 64.0;
    double stuf = 0.0;
```
cout << elligent << endl;
cout << rainbow << endl;

eresting = elligent/2.5;
stuf = rainbow/3;
cout << rainbow/3 << endl;
cout << stuf << endl;
cout << elligent/2.5 << endl;
cout << eresting << endl;
}
64
64
21.3333
21.3333
25.6
25

64 (Printing an integer is simple)
64 (Even if we print a double, because 64 is a whole number, it just prints as 64)
21.3333 (This is a double decimal value.)
21.3333 (This is also a double decimal value.)
25.6 (Even though elligent is an integer, if we use it in an operation, the result will be
a decimal/floating point value!)
25 (This case is different because we are trying to assign a value to an int variable.
Even if the previous result was 25.6, and we're doing the same operation, when we
put that value into an integer variable, the “decimal point” part will get truncated,
leaving the whole number 25.)

When we declare a variable as a certain type, the computer knows to treat that
variable as that type.

Programming Problems

1. Write a program that asks for a number between 0 and 100 (exclusive), and takes
an integer input. If you input a number greater than or equal to 100, it will print “Liar,
liar, plants for hire” and stop. If you input a number less than or equal to 0, it will print
“Liar, liar, plants for hire” and stop. If your number has a “tens” digit that would round
up to 100, meaning the number is at least 50, and will print “Almost to 100!”
Otherwise, it will print “Still a-ways to go!”

**Example Output:**

**Case one:**
Please give me a number between 0 and 100.
0
Liar, liar, plants for hire.
**Case two:**
Please give me a number between 0 and 100.
95
Almost to 100!

**Case three:**
Please give me a number between 0 and 100.
45
Still a-ways to go!

```cpp
int num;

// Print the statement and take the input
cout << "Please give me a number between 0 and 100." << endl;
cin >> num;

// We can check each end in case the number is out of range.
if (num > 99 || num < 1) {
    cout << "Liar, liar, plants for hire." << endl;
} else {
    int tens = num / 10;
    if (tens >= 5) {
        cout << "Almost to 100!" << endl;
    } else {
        cout << "Still a-ways to go!" << endl;
    }
}
```

2. Write a program that takes in two numbers and a command of type string ("Add", "Subtract", "Multiply", "Divide"). Inputting an invalid command should cause the program to print out “Invalid command!” and stop.

Sample output:
Enter your first number: 3
Enter your second number: 7
Enter your command: Multiply
Result: 21

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

int main() {
    int first = 0;
    int second = 0;
    string command = "";

    cout << "Enter your first number: ";
```
cin >> first;

cout << "Enter your second number: ";
cin >> second;
// ignore the next \n to avoid complications!
cin.ignore(1000000, 'n');

cout << "Enter your command: ";
getline(cin, command);

// Check the string to each possible type of command
if (command == "Add")
    cout << "Result: " << first + second << endl;
else if (command == "Subtract")
    cout << "Result: " << first - second << endl;
else if (command == "Multiply")
    cout << "Result: " << first * second << endl;
else if (command == "Divide" && second != 0)
    cout << "Result: " << first / second << endl;
else {
    cout << "Invalid command!" << endl;
}

3. (This may be out of scope for this week, but you can try it as a challenge question!) Write a program that takes in a number as an int and outputs the sum of all of the digits in that number

Sample Output:
Enter a number: 184
The sum of the digits in your number is 13!

#include <iostream>
using namespace std;

int main() {
    cout << "Enter a number: ";
    int num;
    cin >> num;
    int sum = 0;
    /* We want to keep adding until we run out of digits, so we can use a while loop instead of a for loop here */
    while (num > 0) {
        sum += num % 10;
        num /= 10; //or num=num/10;
cout << "The sum of the digits in your number is " << sum << "!" << endl;