PART 1 - From Professor Kent Chin - div_casting.cpp
/* Literals are FIXED values (e.g. 0, 5, -2, 3.14)
   Whole-number literals (e.g. 0, 1, -3) are integer types
   Literals with decimal points (e.g. 3.14, 2.718) are float/double types */
#include <iostream>
using namespace std;

int main() {
    int x = 5; double y = 3;
    // Integer division rounds the result down to the nearest whole number.
    cout << "1a: " << x / 3 << endl; //1

    // Casting converts a variable or literal from one data type to another
    cout << "1b: " << (int)9.73 << endl; //9
    cout << "1c: " << (double)22 / 7 << endl; //3.14286

    // What happens when division between two integers happened BEFORE casting?
    cout << "1d: " << (double)(22 / 7) << endl;  //3

    // Any arithmetic operation with a double value will result in a double
    // value.
    cout << "1e: " << y / x << endl; //0.6
    return 0;
}

PART 2 – Consider the following C++ program.
#include <iostream>
using namespace std;

int main () {
    int i = 7.5; double d = 2;
    cout << i / d << endl;          // line a)
    cout << (double) i / d << endl; // line b)
    cout << i / (int) d << endl;    // line c)
    cout << 7 / 2 << endl;          // line d)
    cout << 15.0 / 4.0 << ' ' << 2.2 + 5 << endl; // line e)
    cout << 5 * 2.0 << endl;        // line f)
    cout << 5 % 3 << endl;          // line g)
    cout << 1 + 4 % 5 * 3 << endl;  // line h)
    return 0;
}

State the output at line:

<p>| | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>a) 3.5</td>
<td>b) 3.5</td>
<td>c) 3</td>
<td>d) 3</td>
</tr>
<tr>
<td>e) 3.75 7.2</td>
<td>f) 10</td>
<td>g) 2</td>
<td>h) 13</td>
</tr>
</tbody>
</table>
Part 3 - Which variable type stores each of the following?

a. single character _______char______  
   b. whole number _______int___________

   c. decimal number ___float or double__  
   d. true or false value _____bool__________

   e. text longer than one character _____string______

Part 4 - Change in bills. (Professor Chin’s hw2)
Write a program that prompts the user for the amount of change in bills that he/she needs. If the user provides a negative amount, print “Invalid amount!” to the screen and exit the program. Otherwise, the program is to print the amount of change (in bills) with hundreds, twenties, tens, fives, and ones (yes, there are other bills out there, but we'll only worry about these bills).

Note: You can use return 0 or exit(1) for program termination. If you are using exit(1), remember to add this to the beginning of the program: #include <cstdlib>

HINT: Feel free to use the “Change In Coins” program from lecture as a reference. And remember to use variable names that make sense.

Sample I/O
How much change in dollars is needed? 999
You need:
9 hundred dollar bill(s)
4 twenty dollar bill(s)
1 ten dollar bill(s)
1 five dollar bill(s)
4 one dollar bill(s)

How much change in dollars is needed? 567
You need:
5 hundred dollar bill(s)
3 twenty dollar bill(s)
0 ten dollar bill(s)
1 five dollar bill(s)
2 one dollar bill(s)

How much change in dollars is needed? -231
Invalid amount!
Part 5 – (Fall’14 MT1) A number is called evil if its last two digits add to 13. Write a complete C++ program that does the following.
1. It asks the user to enter an integer n that is greater than 666.
2. It terminates when given illegal input.
3. It prints out whether n is evil.

Here is an example of how the program should work:
Give me an integer greater than 666: 667
Evil

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

int main() {
    int n;
    cout << "Give me an integer greater than 666:"; cin >> n;
    if (n <= 666) return 0;
    int lastDigit = n % 10;
    n = n / 10;
    int nextLast = n % 10;
    if ((lastDigit + nextLast) == 13) cout << "Evil\n";
    else cout << "Not evil\n";
    return 0;
}
```
Consider the following C++ program. What is the output from the program in response to the following user input?

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

int main() {
    int x, y;
    cout << "Please enter two positive integers: ";
    cin >> x >> y;
    if (x <= 0) x = y;
    if (y <= 0) {
        cout << "Illegal" << endl;
        return 0;
    }
    if (x <= 10) cout << y << x << endl;
    while (y > 0) {
        cout << y;
        y = y / 10;
    }
    cout << x << endl;
    return 0;
}
```

a) The user enters: -5 5

```
55
55
```

b) The user enters: 5 -5

```
Illegal
```

c) The user enters: -5 -5

```
Illegal
```

d) The user enters: 567 123

```
123121567
```

e) The user enters: 567 0

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

int main() {
    int n;
    cout << "Enter a number greater than 0: ";
    cin >> n;
    if (n <= 0) {
        cout << "Illegal.\n";
        return 0;
    }

    while (n > 0) {
        cout << n % 10.0;
        n = n / 10;
    }

    return 0;
}

Bonus: Check out professor Tanasyuk’s extra thought question in http://venus.cs.qc.edu/~mtanasyuk/CS111Lab4Exercise2.pdf