Uninitialized Variables

An **uninitialized variable** is a variable that is declared but is not set to a definite known value before it is used. It will have *some* value, **but not a predictable one.**

A common assumption made is that all variables are set to a known value, such as zero, when they are declared. While this is true for many languages, **it is not true for all of them**, and so the potential for error is there.

C++ is one of the languages that will have an **unpredictable** value for uninitialized variables.

Java, for example, will have predictable values. Java does not have uninitialized variables.

For example: If you look at HW 4 Q3 we went over yesterday in lab

```
#include<iostream>
using namespace std;
int main()
{
        int num, pos, neg;
        double sum, avg;
        cout << num << endl << pos << endl << neg << endl;
        cout << "Enter an int, enter 0 to stop: ";
        cin >> num;
        while (num != 0)
                if(num > 0)
                        pos++;
                else
                        neg++;
                sum += num;
               cin >> num;
"quiz2.cpp" 35L, 413C written
[aabreu@venus ~]$ g++ quiz2.cpp
[aabreu@venus ~]$ ./a.out
4196432
Enter an int, enter 0 to stop:
```

--- I just printed the first three uninitialized variables, (num, pos, neg).

num printed out zero

pos printed out a "garbage" value, 4196432

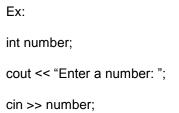
neg printed out zero

This shows us uninitialized variables have unpredictable values.

So when do you have to initialize variables???

When you read from a variable before you write to it

Though, many sources recommend initializing every variable you declare, except if you are assigning it a value in the next couple of lines.



Extra:

If you use a shortcut such as (++), remember you are reading before you are writing and you can end up with an unknown value if the variable is not initialized.

If you recall yesterday we were getting a weird value after we incremented the variable pos.

This is because,

pos++;

is equivalent to

pos = pos + 1;

We are first reading the value from pos, then adding 1 to that value, and finally assigning that new value back to pos.