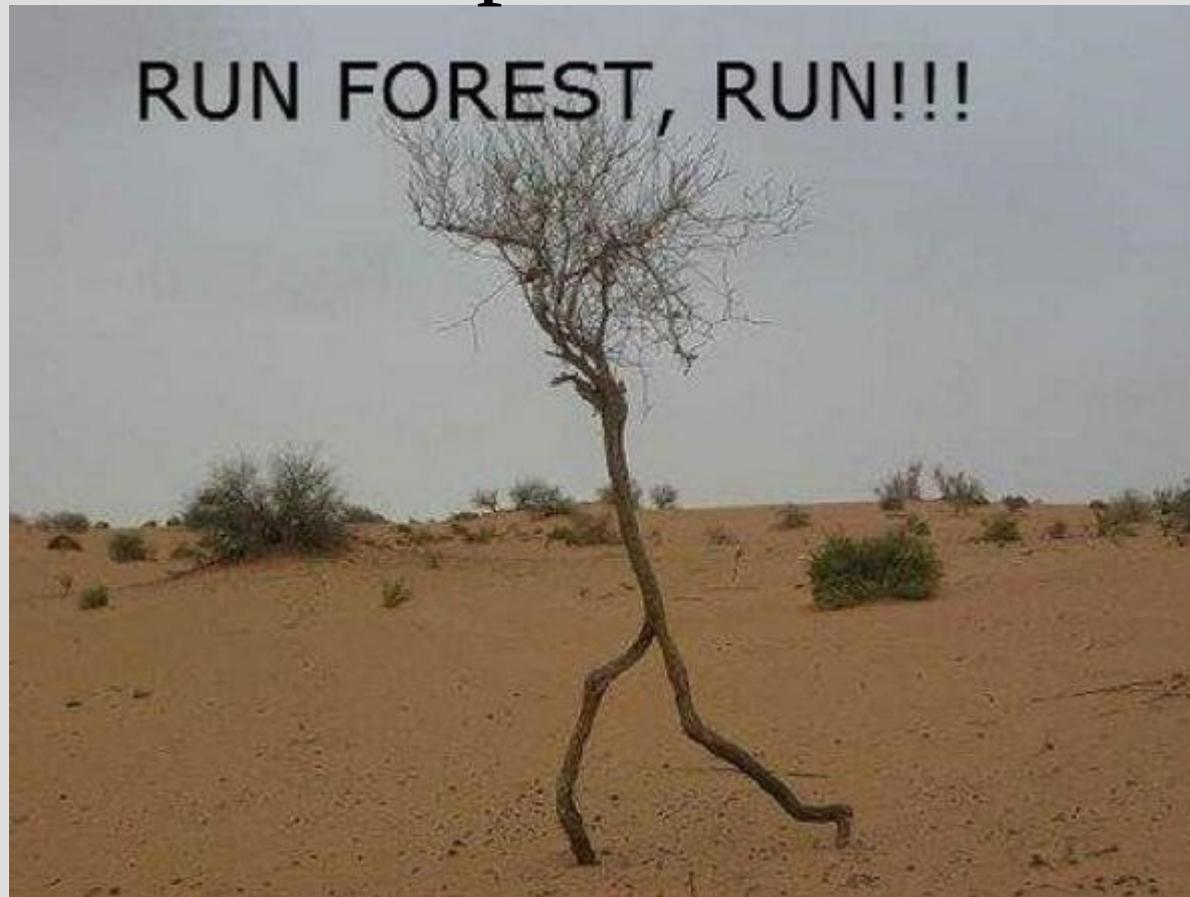```
int getRandomNumber()
{
    return 4;   // chosen by fair dice roll.
                // guaranteed to be random.
}
```

# Functions

How to make the person run?



RUN FOREST, RUN!!!

(See: runForest.cpp)

# Functions

You can also use functions that return
bool types in an if statement or loop

This is commonly used if you have complex
logic as it is normally easier to write
a function that have a very complex
bool expression

(See: findPrime.cpp)

# scope

(See: sillySwap.cpp)

Typically the value of variables is copied and not given access to the real value

This is similar to moodle, the score you see for grades cannot change the score I give you!

# scope

Blocks (inside { }) of code can only see variables from their parent blocks

You can also make global variables outside of all blocks
(almost as if your whole program has a start and end brace around it)

(See: globalVariable.cpp)

# scope

You can give away your memory location by using "call by reference" with functions

This will share the variable between the two functions, namely the function that is using the references (&) can modify the value

```cpp
void realSwap(int &x, int &y)
```

(See: callByReferenceSwap.cpp)

scope

We will talk more about the difference between a variable's <u>memory location</u> and <u>value</u> later

For now, a memory location (or <u>reference</u>) will give a function full access to modify the value

References

When memory does not actually hold the value of an object, but instead holds information about the actual location...

(See: changeInt.java)

... this is called a <u>reference</u>

References

If you use a normal function (call by value) then you will essentially make a photo copy of the variables
   (makes 2 variables, does not effect other)

If you use call-by-reference, you have only one variable, but share it between you two

This is similar to a website link, if two people follow the link they end up in the same page

Debugging

-Test small pieces of code at a time
-Add cout statements to see values in loops (and to localize error in general)
-Test code on inputs you know the answer