STATIC VARIABLES

Scope of Variables VS Their Lifetime:

Scope and lifetime usually coincide, but ***static variables persist even after they are out of scope - program lifetime!***

Using the `static` keyword on local variables changes them to static duration. A static (duration) variable is one that retains its value even after the scope in which it has been created has been exited.

Static variables are only created and initialized once, and then they persist throughout the execution of the program.

```
void incrementAndPrint()
{
    int value = 1; // automatic duration
    ++value;
    std::cout << value << '\n';
} // value is destroyed here

int main()
{
    incrementAndPrint();
    incrementAndPrint();
    incrementAndPrint();
}
```

```
>> 2
>> 2
>> 2
```

```
void incrementAndPrint() {
    static int s_value = 1; // static duration
    ++value;
    std::cout << value << '\n';
} // s_value is not destroyed here,
  // but becomes inaccessible

int main() {
    incrementAndPrint();
    incrementAndPrint();
    incrementAndPrint();
}
```

```
>> 2
>> 3
>> 4
```

One of the most common uses for static duration local variables is for unique identifier generators.

When dealing with a large number of similar objects within a program, it is often useful to assign each one a unique ID.

This is very easy to do with a static duration local variable:

```
int generateID() {
    static int s_itemID = 0;
    return s_itemID++;  //increments
      //itemID and returns new value
}
```

The first time this function is called, it returns 0

The second time, it returns 1

Each time it is called, it returns a number one higher than the previous time it was called

You can assign these numbers as unique IDs for your objects

However, because `itemID` is a ***local variable***, it can not be “tampered with” by other functions.

Static variables offer some of the benefit of global variables:

they do not get destroyed until the end of the program…

...while limiting their visibility to ***block scope***

This makes them ***much safer*** for use than global variables.