Name: ____

Today's Date: ____________________________________

Today's Goals:

- List the stages of a primitive, local variable's lifetime.
- Name 2 ways a primitive can be initialized.
- Use C++ arrays

Today's Questions

What is a *compiler*'s job?

How is Homework 1 going for you?

Lingering Questions:

What is still unclear after today's class?

Object lifetimes

Every object goes through these stages, over the course of its life.

- **Allocation**: acquire memory for the object
- **Initialization**: create the object
- **Use**: access the object
- **Destruction**: destroy the object
- **Deallocation**: relinquish the object's memory

Object Lifetimes for Local Variables: When?

- Allocation:
- Initialization:
- Use:
- Destruction:
- Deallocation:

Functions and Local variables

Functions manage the lifetimes of their *local variables*. In CS70, a function's local variables are:

Function's Perspective

- At the opening {, ...
- During a function, for each line of code, ...
- At the end of a block, ...
- At the end of the function, ...

Exercises

What is an array?

Why do we have arrays?

(After Homework 4, we'll have `vector`)

C++ primitive arrays

```
// read this declaration "inside out"
int values[42]
```

~~int values[]~~;

```
// lifecycle rules apply
int weeklyPayments[DAYS_IN_WEEK];
```

```
const int weeklyPayments[DAYS_IN_WEEK];
```

```
// initialization
int weeklyPayments[DAYS_IN_WEEK] = {10,5,5,5,5,5,10};
```

Array Idiom

It's okay to default initialize the elements of an array, if we then *immediately* initialize *all* the elements.

```
int weeklyPayments[DAYS_IN_WEEK];

for (size_t day = 0; day < DAYS_IN_WEEK; ++day) {
   cin >> weeklyPayments[day];
}
```

C++ primitive arrays: indexing

```
int weeklyPayments[DAYS_IN_WEEK] = {10,5,5,5,5,5,10};
cout << weeklyPayments[0] << endl;
cout << weeklyPayments[1] << endl;
```

What happens if we write:

```
int values[3] = {1, 2, 3};
cout << values[10000] << endl;
```

Looking Forward

- Grutoring is up and running!
- Homework 1 is due Wednesday night
- Homework 2 (data visualization with embroidery) is available Thursday

Exercise 1

```
int triple(int multiplier)
{
   int product = 3 * multiplier;
   return product;
}

int main()
{
    int myInt = 0;
    int myConstant = 14;
    myInt = triple(myConstant);

    cout << myInt << endl;

    return 0;
}
```

Exercise 2

```
int cube(int base)
{
    int outcome = base * base;
    outcome = outcome * base;
    return outcome;
}

int main()
{
    int myInt = 0;
    int myConstant = 3;
    myInt = cube(myConstant);

    cout << myInt << endl;

    return 0;
}
```