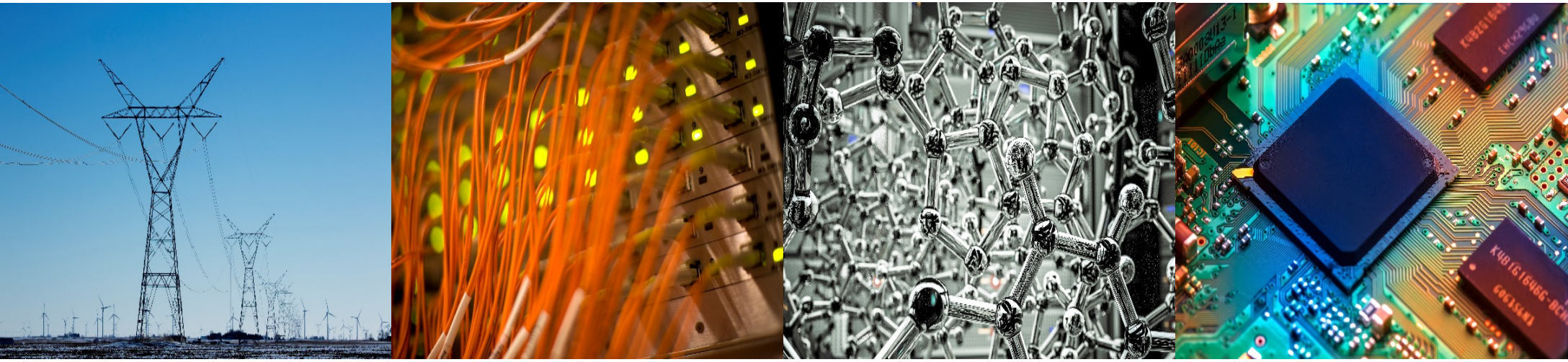
ECE 220 Computer Systems & Programming

Lecture 31 – C++ Class & Encapsulation

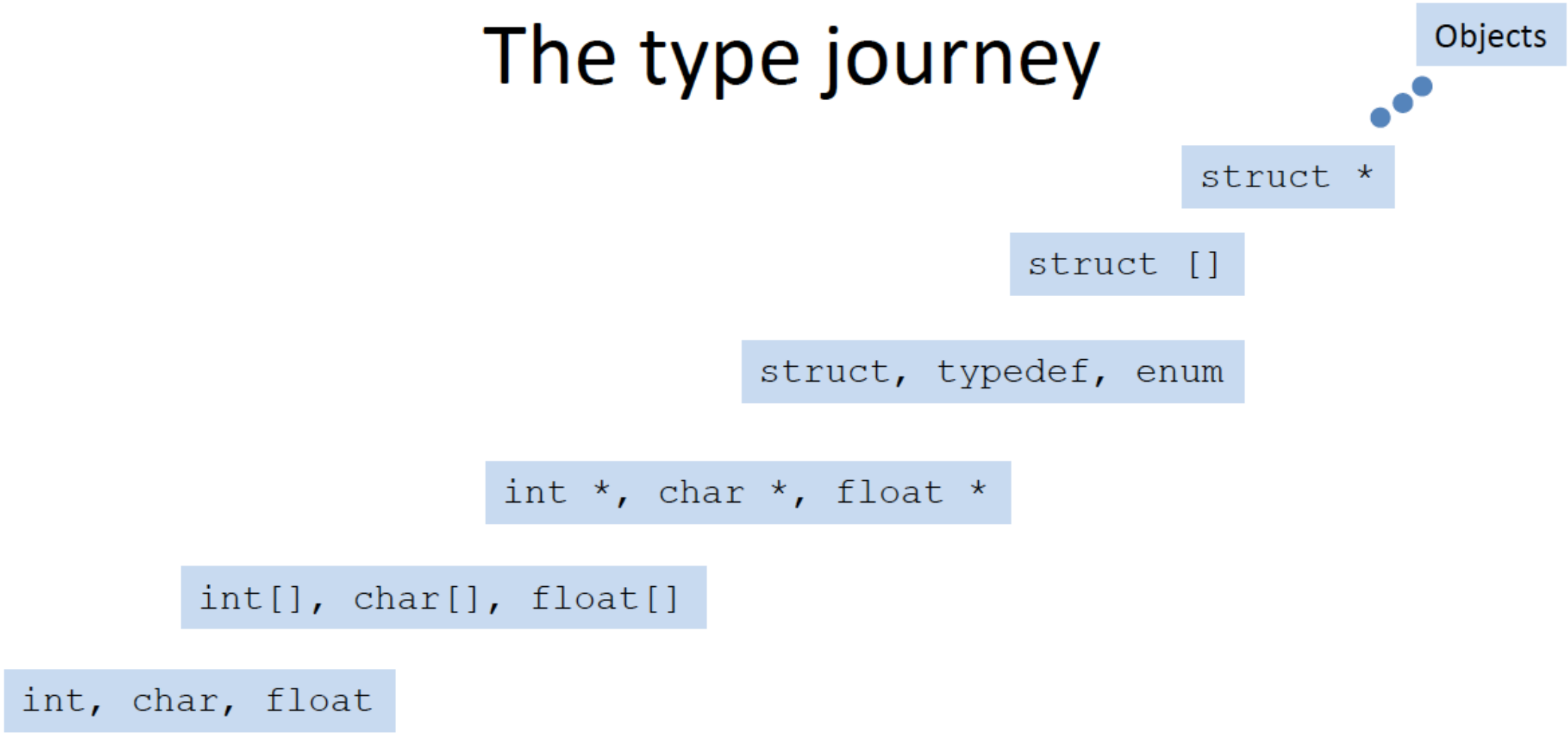
July 30, 2020



ILLINOIS
Electrical & Computer Engineering
GRAINGER COLLEGE OF ENGINEERING

- Midterm 2 score posted on Gradescope
- Regrade request deadline: 10pm on Friday, 7/31
- MP7 & MP8 due next Wednesday

The type journey



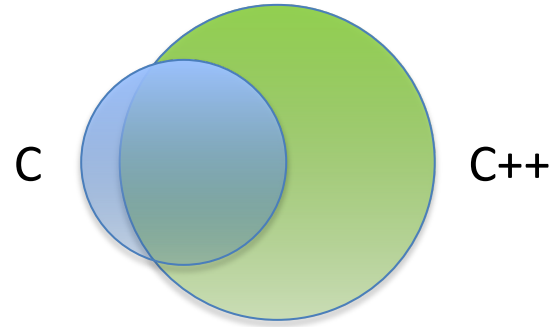
C++ – Class & Encapsulation

- Created in 1979 by Bjarne Stroustrup at Bell Labs, as an extension to C
- It's an **object-oriented** language

  OOP Concepts:

  Encapsulation, Inheritance

  Polymorphism, Abstraction



Class in C++ is similar to Struct in C, except it defines the data structure **AND**

- control "who" can access that data
- provide functions specific to the class

➢ Spot the differences in C vs. C++ examples for adding two vectors

Concepts Related to Class

An **object** is an instance of the class

- shares the same functions with other objects of the same class
- but each object has its own copy of the data

member functions (also called methods) - functions that are part of a class

Private vs. Public members

- private members can only be accessed by member functions (private access is the **default** in a class)
- public members can be accessed by anyone

Constructors & Destructors

- Constructor – a special member function that *creates* (initiates) a new object
- Destructor – a special member function that *deletes* an object (when it goes outside of scope)

Basic Input / Output

cin – standard input stream
cout – standard output stream

namespace –
"using namespace" directive tells compiler the subsequent code is using names in a specific namespace

Example:

```
#include <iostream>
using namespace std;
int main(){
      char name[20];
      cout << "Enter your name: ";
      cin >> name;
      cout << "Your name is: " << name << endl;
}
```

Exercise – Write Constructors

```
class Rectangle{
      int width, height;
    public:
      Rectangle();
      Rectangle(int, int);
      int area() {return width*height;}
};
Rectangle::Rectangle(){
//set both width and height to 1

}
Rectangle::Rectangle(int a, int b){
//set width to a and height to b


}
```

Exercise – Access Member in a Class

```
#include <iostream>
int main(){
      Rectangle rect1(3,4);
      Rectangle rect2;

      //print rect1's area


      //print rect2's area


      return 0;
}
```

- What is the area of object rect1? How about rect2?
- How do we get the width/height of each object?

Dynamic Memory Allocation

new – operator to *allocate* memory (similar to *malloc* in C)

delete – operator to *deallocate* memory (similar to *free* in C)

Example:

```
int *ptr;
ptr = new int;
delete ptr;

int *ptr;
ptr = new int[10];
delete [] ptr;
```

Exercise – Pointer to an Object

```
int main(){
        Rectangle rect1(3,4);
        Rectangle *r_ptr1 = &rect1;
        //print rect1's area through r_ptr1

        Rectangle *r_ptr2, *r_ptr3;
        r_ptr2 = new Rectangle(5,6);
        //print area of rectangle pointed to by r_ptr2

        r_ptr3 = new Rectangle[2]{Rectangle(),Rectangle(2,4)};
        //print area of the 2 rectangles in the array


        //deallocate memory


        return 0;
}
```