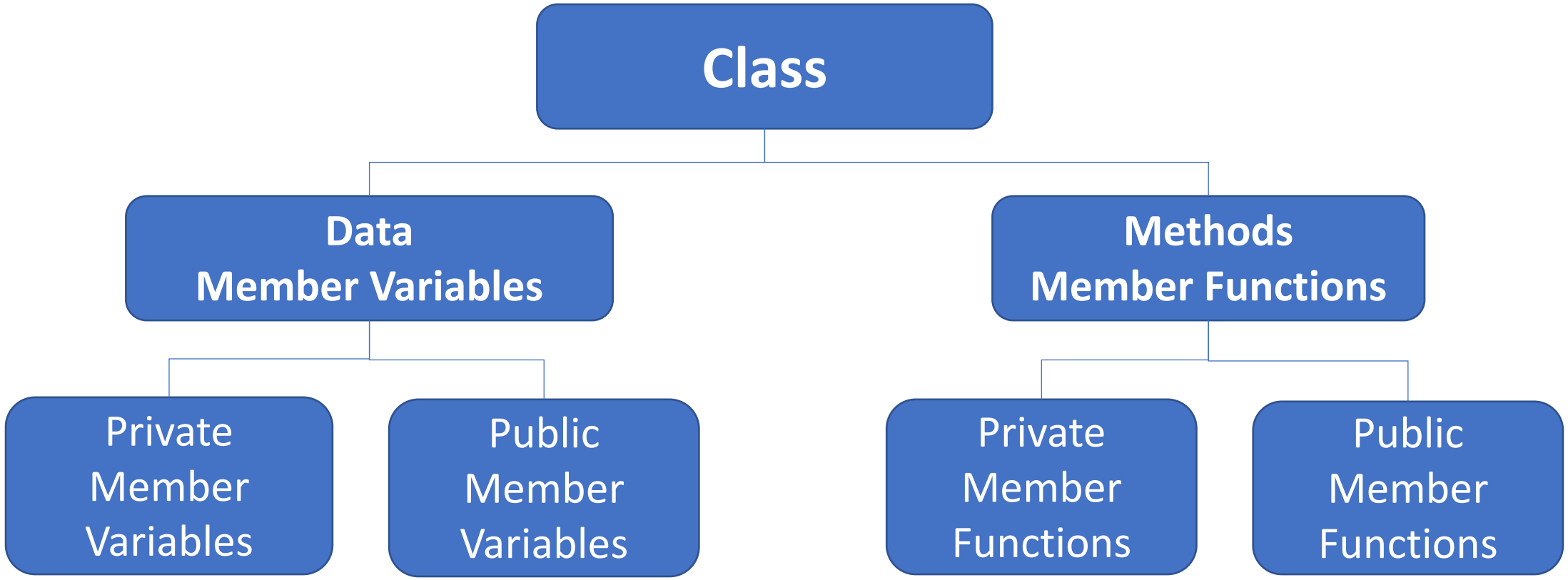
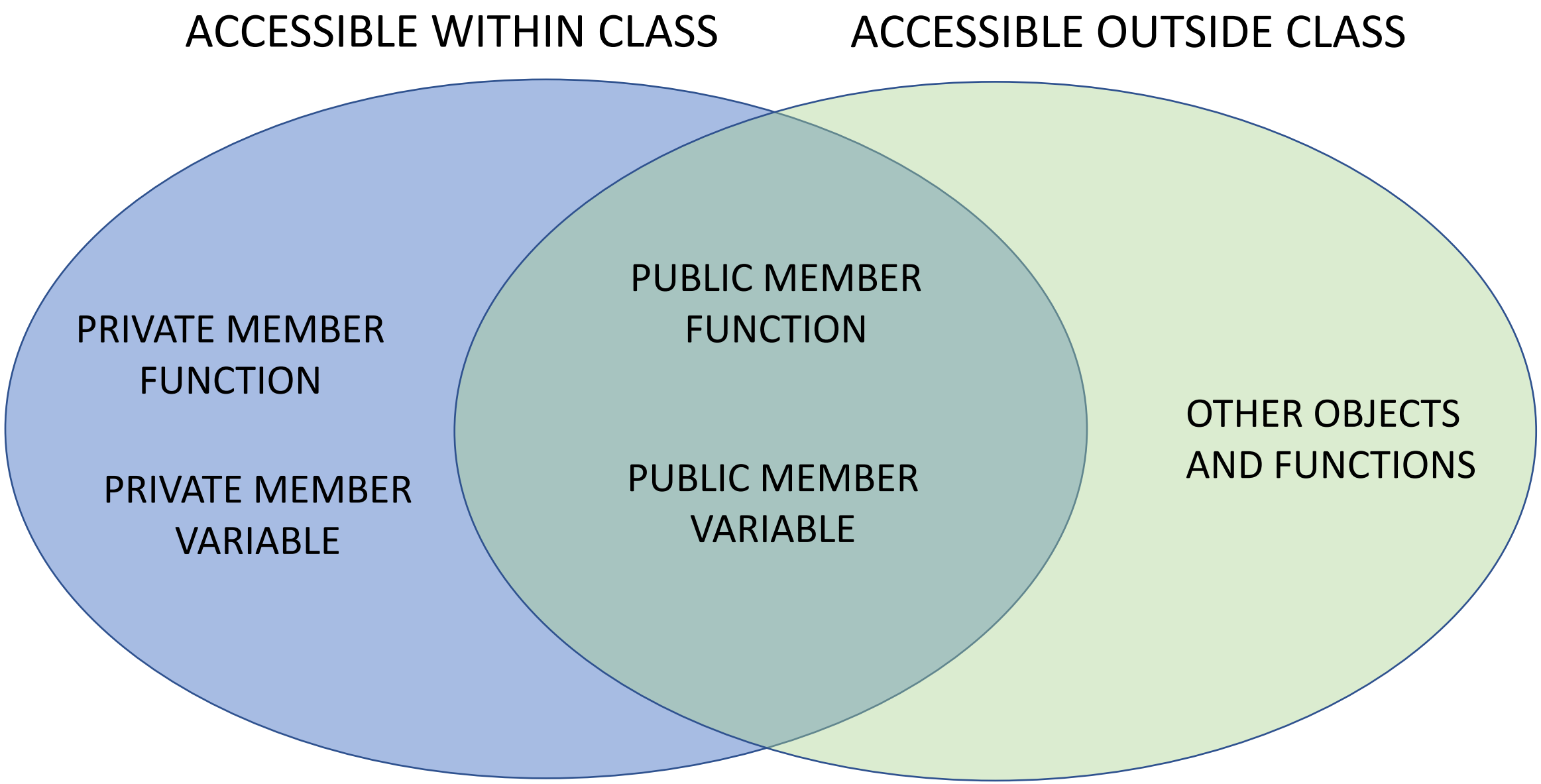
C++ Classes

02/12/2020

Classes are a basic building block of c++ code

- C++ is object oriented code
  - Everything we do is defining and manipulating “**objects**”
- Objects are either
  - **data structures** like int, str, float…
  - or **user-defined classes**
- An instance of a class is an object


Class
Data
Member Variables
Methods
Member Functions
Private
Member
Variables
Public
Member
Variables
Private
Member
Functions
Public
Member
Functions


ACCESSIBLE WITHIN CLASS
ACCESSIBLE OUTSIDE CLASS
PRIVATE MEMBER FUNCTION
PRIVATE MEMBER VARIABLE
PUBLIC MEMBER FUNCTION
PUBLIC MEMBER VARIABLE
OTHER OBJECTS AND FUNCTIONS

Defining a Class

```
Class ClassName
{
        public:                 //access specifier private, public
        int some_var;
        type member_function(){
                std::cout<<other_var<<std::endl;
        }

        private:
        double other_var = 5.2;

        // constructor
        ClassName(int j): some_var(j){
        //some code here
        }
};
```

```
void main(){
    ClassName sample(3);
    std::cout<<sample.some_var<<std::endl;
    sample.member_function();

    return;
}
```

```
>> ./main
3
5.2
```

```
class Vehicle
{
        public:
                int wheels;
                double speed;
                void set_speed(double v){
                        speed=v;
                        return;}
                double get_speed(){ return speed;}
                int wheel_num(){return wheels;}

//default constructor:
                Vehicle(){
                        speed=0.0; wheels=4;
                }
//Parameterized constructor
                Vehicle(int w, double s) {
                        wheels=w;  speed=s;
                }
};
```

Constructor examples

```
//default constructor:
        Vehicle(){speed=0.0; wheels=4;}

//Parameterized constructor
        Vehicle(int w, double s): wheels(w),  speed(s) {}

//another Parameterized Constructor
        Vehicle(int w, double s) {wheels=w;  speed=s;}

//Parameterized with default values:
        Vehicle(int w=4, double s=0.0): wheels(w),  speed(s) {}
```

Constructor examples

```
//default constructor:
        Vehicle(){speed=0.0; wheels=4;}

//Parameterized constructor
        Vehicle(int w, double s): wheels(w),  speed(s) {}

//another Parameterized Constructor
        Vehicle(int w, double s) {wheels=w;  speed=s;}

//Parameterized with default values:
        Vehicle(int w=4, double s=0.0): wheels(w),  speed(s) {}
```

```
Vehicle car();
car.get_speed();        returns-> 0.0
car.wheel_num();        returns-> 4
```

Constructor examples

```
//default constructor:
        Vehicle(){speed=0.0; wheels=4;}

//Parameterized constructor
        Vehicle(int w, double s): wheels(w),  speed(s) {}

//another Parameterized Constructor
        Vehicle(int w, double s) {wheels=w;  speed=s;}

//Parameterized with default values:
        Vehicle(int w=4, double s=0.0): wheels(w),  speed(s) {}
```

```
Vehicle car();          → behaves badly

Vehicle.tricycle(3,1.2);

tricycle.get_speed();          returns-> 1.2
tricycle.wheel_num();          returns-> 3
```

Constructor examples

```
//default constructor:
        Vehicle(){speed=0.0; wheels=4;}

//Parameterized constructor
        Vehicle(int w, double s): wheels(w),  speed(s) {}

//another Parameterized Constructor
        Vehicle(int w, double s) {wheels=w;  speed=s;}

//Parameterized with default values:
        Vehicle(int w=4, double s=0.0): wheels(w),  speed(s) {}

//To throw and error when initialized incorrectly change to:
        explicit Vehicle(int w=4, double s=0.0): wheels(w),  speed(s) {}
```

```
Vehicle.car();
Vehicle.bicycle(2,5.4);
Vehicle.unicycle(1);

car.get_speed();          returns->0.0
bicylce.get_speed();      returns ->5.4
unicycle.get_speed();     returns-> 0.0

car.set_speed(25.2);
car.get_speed();          returns->25.2;

Vehicle.truck(34.2);      →bad things happen
```