Supplemental Lab: Inheritance

Jamie Flores

Today's Schedule

1. What?
2. Why?
3. How?
4. Example
5. Assignment!

What is inheritance?

Create new classes from existing ones that are related.

Inherits from

Person

Student

1

Why inheritance?

We naturally think in categories and subcategories!

Why use inheritance?

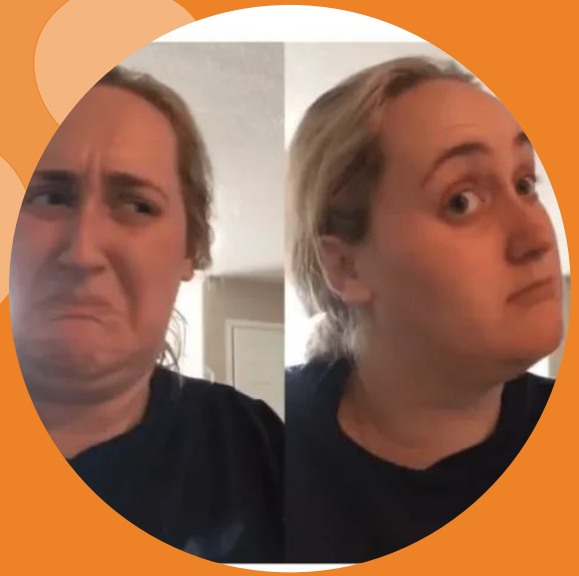
Reuse functions and data members

string name;
int age;

void thumbsUp();

Person

Student



How?

- Syntax
- Constructors
- Private/Protected/Public
- Virtual/Abstract
- Destructors

Syntax (header file)

```
class Person {
    public:
        Person(std::string name);
        std::string getName();
    private:
        std::string mName;
        int mAge;
};

class Professor : public Person {
    public:
        Professor(std::string name, std::string department);
        std::string getDepartment();
    private:
        int mSalary;
        std::string mDepartment;
};
```

Parent

Child

Constructors

```
Student::Student(string name) : Person(name) {
    // rest of student constructor
}
```

Gets called BEFORE the rest of the constructor!

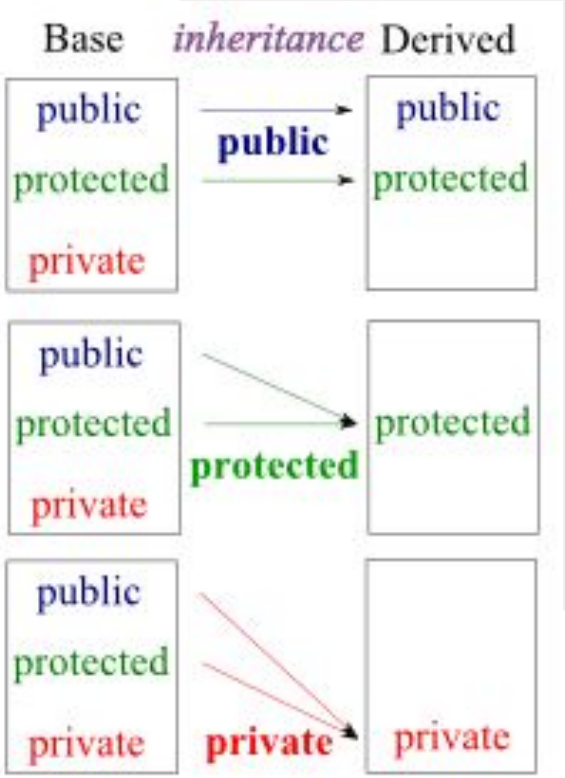
Private/Protected/Public Data Members

1. **Private**: child classes cannot access

2. **Protected**: child classes can, but not 3rd parties

3. **Public**: anyone can access

Private/Protected/Public Inheritance



Private/Protected/Public Example

- Which functions can class A access?
  - B?
  - C?
- Which functions can a user of A access?
  - User of B?
  - User of C?

```
class A {
    public:
        void funcA();
};

class B : private A {
    protected:
        void funcB();
};

class C : public B {
    public:
        void funcC();
};
```

Private/Protected/Public Example

- Which functions can class A access? funcA()
  - B? funcA() AND funcB()
  - C? funcB() AND funcC()
- Which functions can a user of A access? funcA()
  - User of B? Nothing!
  - User of C? funcC()

```
class A {
    public:
        void funcA();
};

class B : private A {
    protected:
        void funcB();
};

class C : public B {
    public:
        void funcC();
};
```

Virtual (good exam question alert!)

- Compile time vs. Runtime type

```
Person* p = new Professor();
p->printTitle();
```

```
class Person {
  public:
    virtual void printTitle(); // prints "Person"
};

class Professor : public Person {
  public:
    void printTitle(); // prints "Professor"
};
```

- Virtual functions go by runtime type!

Destructors should be virtual!

- Why?

  MEMORY LEAKS!

```
class Person {
    public:
        Person();
        ~Person();
    private:
        std::string name;
};

class Student : public Person {
    public:
        Student();
        ~Student();
    private:
        // Dynamically allocated array of courses.
        std::string* courses;
};

Person* p = new Student();
delete p; // <--- which deconstructor will get called here?
```

Example!

Let’s handle error messages together

2

Assignment!

NOTE: for Inventory, the items in the list are supposed to be *unique*. They have associated amounts.

Yay Inheritance!

Any questions? Ask on Piazza in this thread!

Pause this video and complete the assignment.