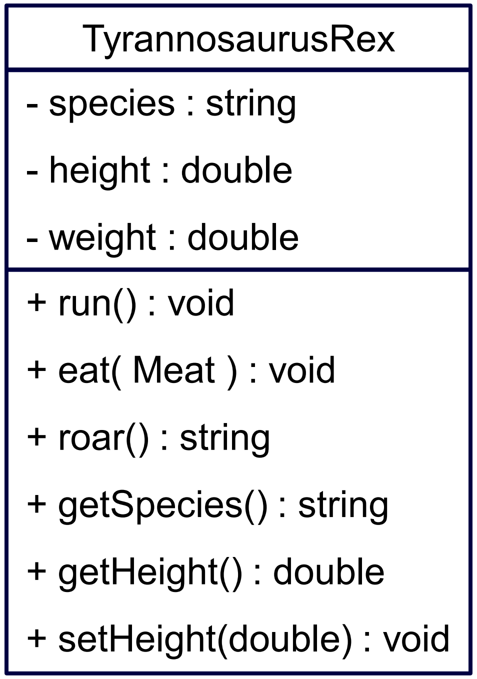
CSCI 200: Foundational Programming Concepts & Design Lecture 19

Collections of Objects

Previously in CSCI 200

- Use + - to denote public private



```
TyrannosaurusRex
- species : string
- height : double
- weight : double
+ run() : void
+ eat( Meat ) : void
+ roar() : string
+ getSpecies() : string
+ getHeight() : double
+ setHeight(double) : void
```

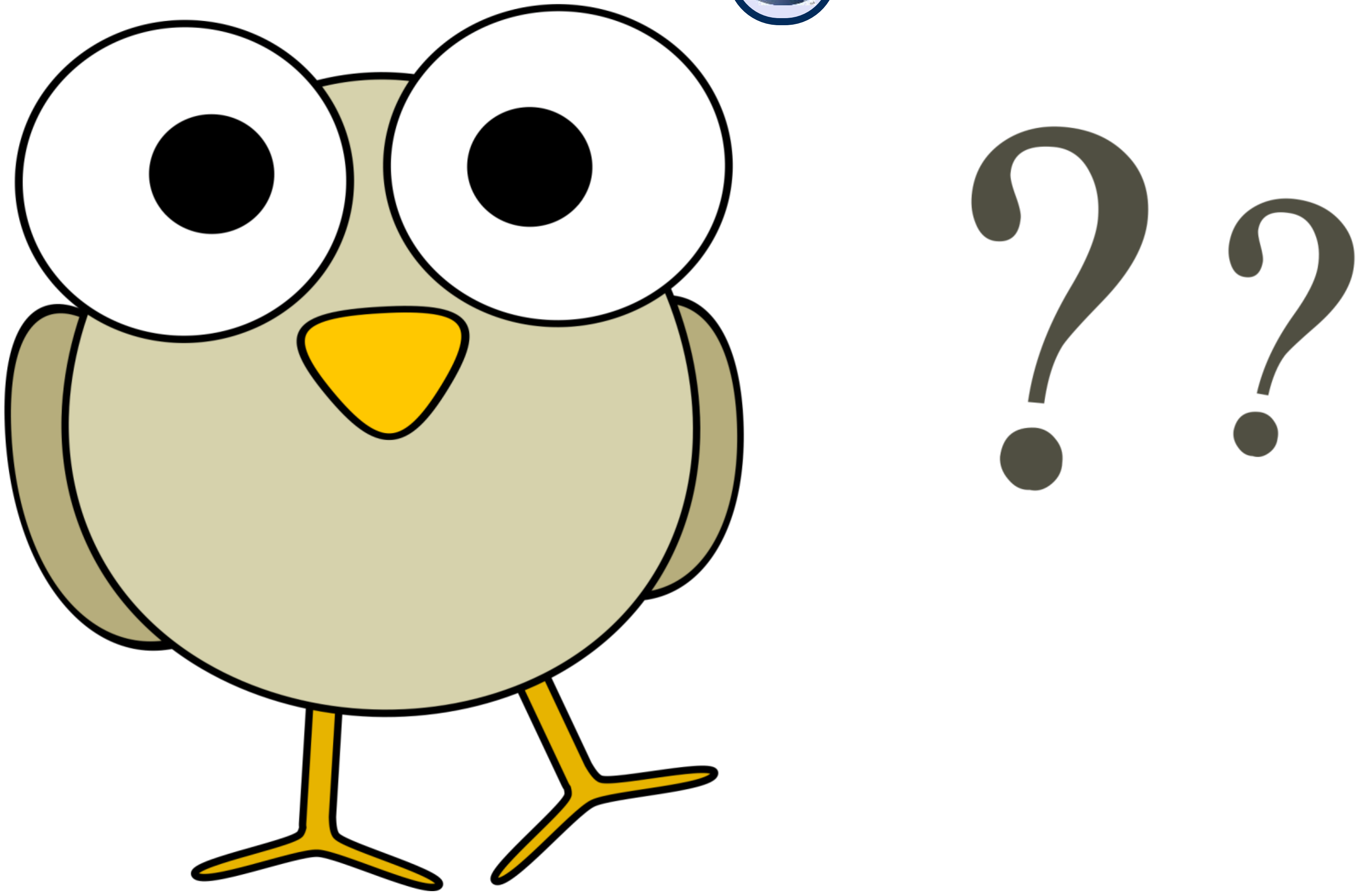
Previously in CSCI 200

```cpp
// inside Box.h
class Box {
public:
    Box();
    Box( int h, int w, int d );
    int volume();
    int getHeight();
    void setHeight(const int H);
    // others for width & depth
private:
    int _height;
    int _width;
    int _depth;
};
```

```cpp
// inside Box.cpp
#include "Box.h"

int Box::getHeight() {
    return _height;
}


void Box::setHeight(const int H) {
  if(H > 0) _height = H;
}


// others for width & depth
```

```cpp
// main.cpp
Box myBox(5, 5, 5);
cout << myBox.volume() << endl; // 125

myBox.setWidth(-5);
cout << myBox.volume() << endl; // 125

myBox.setHeight(10);
cout << myBox.volume() << endl; // 250
```

Questions?



Learning Outcomes For Today

- Construct a program that accesses an element in a vector, returns the length of a vector, changes the length of the vector, and other vector operations.
- Construct a program that accesses an element in a string, returns the length of a string, changes the length of the string, and other string operations.
- Compare and contrast Procedural Programming with Object-Oriented Programming
- Explain the following terms and how they are used (1) dot operator / member access operator (2) data member (3) scope resolution operator
- Discuss the concept of scope within and outside a class & struct

On Tap For Today

- Collections of Objects
- Final Project
- Practice

On Tap For Today

- Collections of Objects
- Final Project
- Practice

To Do: Create this Class

- Create a vector of Courses: populate and print contents

```
Course
- enrollment: int                  // initializes to zero
- title: string                    // initializes to CSM101
+ Course()
+ Course(string)                   // sets title to param
+ getTitle(): string               // returns title of course
+ getEnrollment(): int             // returns enrollment of course
+ registerStudent(): void          // increments enrollment by 1
+ withdrawStudent(): void          // decrements enrollment by 1
```

- Will submit source files to Canvas under Lecture 19 In Class Activity

Sample Class

```
class Course {
public:
  Course() {
        _enrollment = 0;
        _title = "CSM 101";
  }
  Course(const string TITLE) {
        _enrollment = 0;
        _title = TITLE;
  }
  string getTitle() { return _title; }
  int getEnrollment() { return _enrollment; }
  void registerStudent() { _enrollment++; }
  void withdrawStudent() { if(_enrollment > 0) _enrollment--; }
private:
  int _enrollment;
  string _title;
};
```

To Do: Create this Class

- Submit your files zipped together to canvas Lecture 19 In Class Activity (if you haven’t done so yet)

```
             Course
---------------------------------
- enrollment: int                 // initializes to zero
- title: string                   // initializes to CSM101
---------------------------------
+ Course()
+ Course(string)                  // sets title to param
+ getTitle(): string              // returns title of course
+ getEnrollment(): int            // returns enrollment of course
+ registerStudent(): void         // increments enrollment by 1
+ withdrawStudent(): void         // decrements enrollment by 1
```

On Tap For Today

- Collections of Objects
- Final Project
- Practice

Final Project

- Requirements
  - F Oct 17 - PDF to Canvas
    - Project Proposal
  - R Dec 11- zip to Canvas
    - Project Code
    - Project Paper

- Note: R Dec 11 - last day for all submissions!

Project Proposal

- Title
- Program Description (one paragraph)
- Data Description
  - Class UML Diagrams (Pseudocode) → NO code
  - List Data Structure
  - File I/O
- Procedural Description
  - Pseudocode → NO code
- Concerns / Needs

Project Code

- Has at least one original class
  - Private data/functions
  - Well defined Public interface
  - Written in separate files
  - More functionality than just getters/setters
- Has at least one array/linked list/queue/stack
  - List within class OR list of objects in main
- Uses File I/O
- Uses functions & constants when appropriate
- LOTS of comments
- Must follow style guidelines & best practices

Project Paper

- Title
- Program Description (one paragraph)
- Documentation
  - How to run and use your program
- Why was class structured as such?
- Why was data structure chosen?
- Why was File I/O used?
- What changed and why?
- Reflections

On Tap For Today

- Collections of Objects
- Final Project
- Practice

To Do For Next Time

- Wednesday: File I/O + vector & string Quiz
- Friday: Final Project Proposal due
- Fall Break!
- Submit class code to Canvas now