Homework 2 Overview

- Your “empty” object code from Homework 1 should be pushed to your repo!
- Data Flow Table
  - Build on the Scenario Walkthrough table you filled out last week
  - Go through the game spec and game scenarios
  - For each task involving your object, list data you need to receive and return
  - Goal: plan the methods you need to implement and those you need to call
  - Bonus: make sure your teammates are preparing the methods you need to call!

Data Flow

- Your object is probably only directly connected with the Game Control object (there are some exceptions though)
- There are still *many* data flows that come from/to various objects *through* the Game Control object

Data Flows for GameLocations			
Task	Called by or Calls This Object	Receives this Data	Returns this Data
Has Player encountered Wumpus?	Called by Game Control	Current room of player	Yes or No
Move Player	Called by Game Control	New room	True if player can be moved
Move Player	Calls Cave	Current room	Array of connected rooms

Stub Class

- Stubbed Class
  - Turn your Data Flow Table into code!
  - Only method definitions
    - Method name
    - Return value
    - Parameters
  - No implementation yet
    - If you need to return something, just use a fixed value
- Submit the code to your team repo!

Homework Overview - Example

- A stub is a method prototype with minimal implementation
- Stubbed class is a class full of stubs
- Example of a “stub”

```
export class GameLocations
{
    // This method will change the player's location.
    // Inputs: room – The new location (an integer)
    // Return: boolean – Was the location valid?
    public movePlayer(room)
    {
        // Code will be added here later
        return true;
    }
}
```

- Your homework assignment must compile successfully, even though it won’t “do anything” yet

Why do we use stubs?

- Define a “contract” about how code works
- Your teammates can call your code when you haven’t written it yet
- In professional coding we use an “interface” for this
  (this is what C#/Java looks like)

```
interface IGameLocations
{
    bool MovePlayer(int room);
}

class MyGLClass : public IGameLocations
{
    MyGLClass() { … code … }

    bool MovePlayer(int room)
    {
        … code …
    }
}
```