# CSE 12: Week 2 Discussion

Focus: Interfaces

# Reminders

- PA1 due tomorrow (Jan 13th) @ 11:59 PM
- PA2 released Thursday morning (~8am), first **closed** assignment

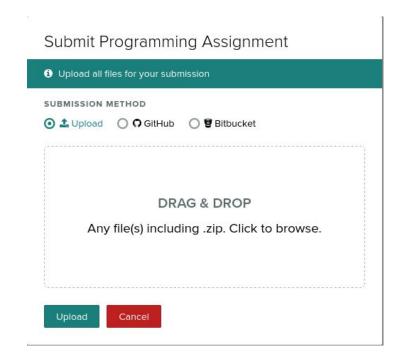
# Submitting to Gradescope

Each programming assignment will have two separate Gradescope assignments to submit to i.e.

- "Programming Assignment 1 code"
- "Programming Assignment 1 questions"

The writeup will specify which files to upload. Only submit those files! You may encounter errors if you do not. We suggest using a zip file.

Once you submit, you will see the autograded results. If there is a compile error it will say so. Please read the error messages before seeking help from a tutor/TA



#### Interface

- Contains fields and method signatures only. Classes implementing the interface must override and define these methods.
- To define, use keyword "interface".
- To implement, use keyword "implements".
- Allows for abstraction

```
public interface User{
     boolean canAccess(String assignmentID);
     ...
}
public class Instructor implements User{
     boolean canAccess(String assignmentID){
          ...
}
     ...
}
```

Which one of these is a compile error?

- A. Student s1 = new Student("Andrew");
- B. User u1 = new Student("Jane");
- C. Student s2 = new User("Rachel");

#### Answer - C

An interface cannot be initialized on its own. Interfaces are utilized by implementing them through other concrete classes. So you cannot initialize User since it is an interface.

#### Worksheet

boolean canAccess(String assignmentId) - A method that returns a boolean indicating whether the user can access an assignment.

- Instructors can access all assignments
- Students can only access assignments from their own list

void printAssignments(User u, String[] allAssignments) - A method that loops through all assignments and prints the boolean value of canAccess of the User for each assignment.

## Worksheet solutions - blank 1.1

```
public boolean canAccess(String assignmentId) {
    for (int i = 0; i < this.assignments.length; i++) {</pre>
         if (this.assignments[i].equals(assignmentId)) {
             return true;
    return false;
```

# Worksheet solutions - blank 1.2

```
public boolean canAccess(String assignmentId) {
    return true;
}
```

# Worksheet solutions - blank 1.3

```
public static void printAssignments(User u, String[] allAssignments)
{
    for (int i = 0; i < allAssignments.length; i++) {
        boolean access = u.canAccess(allAssignments[i]);
        System.out.println(access);
    }
}</pre>
```

## Interfaces

Allows code reuse by having multiple classes that share a type (subtype polymorphism).