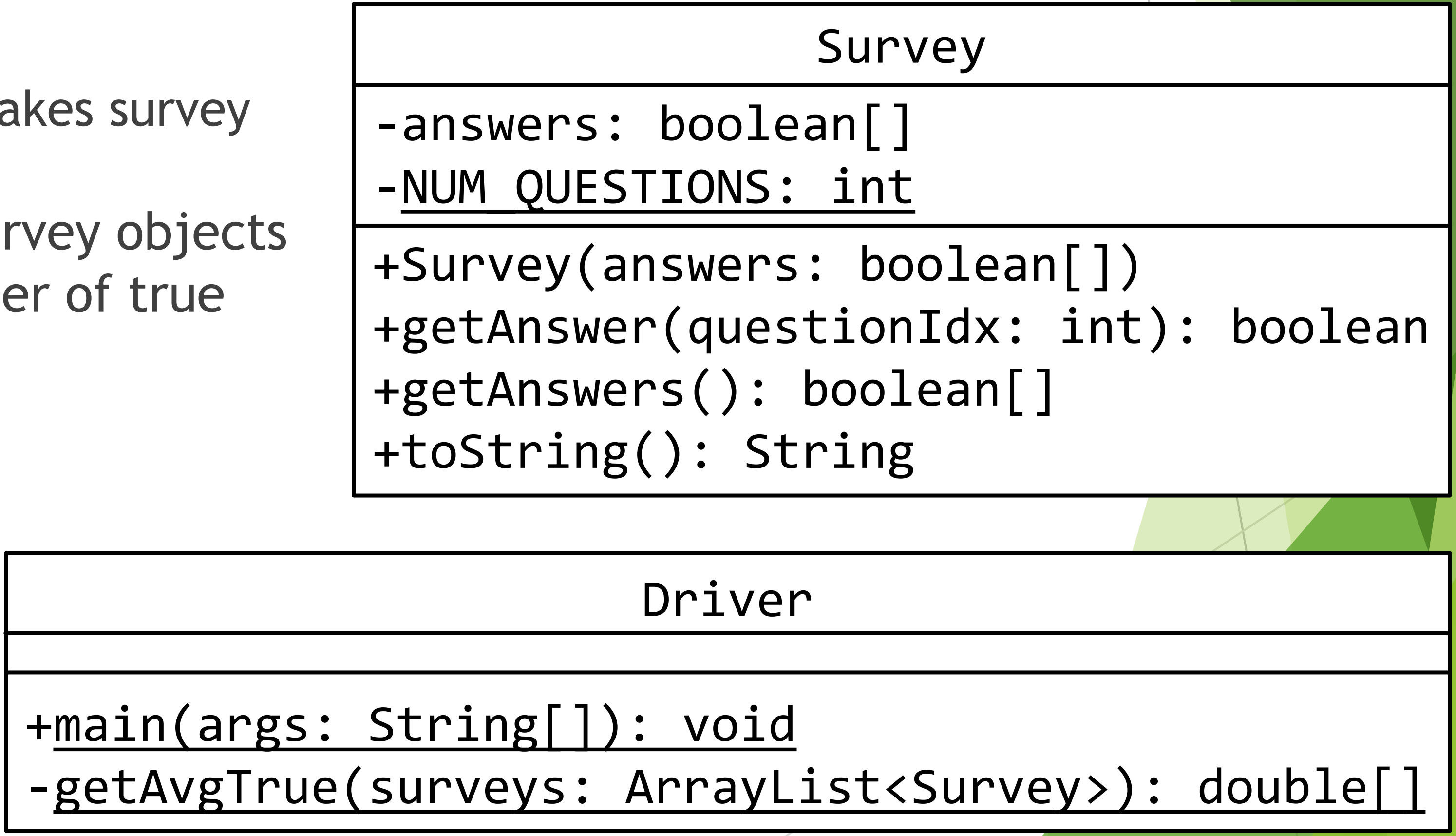
Encapsulation and Aggregation

CS 1323/1324

Example: Survey Program

- Objects of Survey class store the answers to a survey of true/false questions
  - Each survey has the same number of questions: NUM_QUESTIONS (class data)
  - Objects are immutable
  - One object per person that takes survey
- Driver class keeps a list of Survey objects and calculates average number of true answers to each question



```
Survey
-answers: boolean[]
-NUM_QUESTIONS: int
+Survey(answers: boolean[])
+getAnswer(questionIdx: int): boolean
+getAnswers(): boolean[]
+toString(): String
```

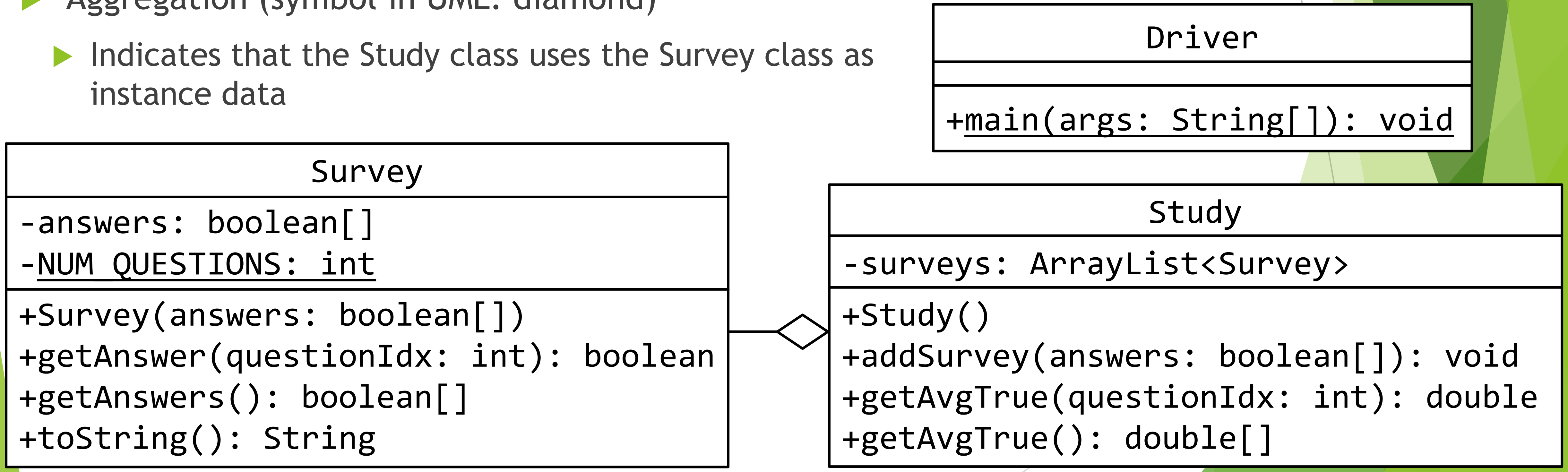
```
Driver
+main(args: String[]): void
-getAvgTrue(surveys: ArrayList<Survey>): double[]
```

Observations

- Driver class is serving two purposes:
  1. Keeping track of the surveys
  2. Handling user input
- Having one class perform two tasks is an encapsulation failure
- One program cannot easily run multiple sets of surveys
  - Although the program can be run twice

Improve the Design with Aggregation

- Have the Driver class only handle user input
- Create a third class to keep track of the surveys
  - Each Study object stores a list of Survey objects for a single study
- Aggregation (symbol in UML: diamond)
  - Indicates that the Study class uses the Survey class as instance data



```
Driver
+main(args: String[]): void
```

```
Survey
-answers: boolean[]
-NUM_QUESTIONS: int
+Survey(answers: boolean[])
+getAnswer(questionIdx: int): boolean
+getAnswers(): boolean[]
+toString(): String
```

```
Study
-surveys: ArrayList<Survey>
+Study()
+addSurvey(answers: boolean[]): void
+getAvgTrue(questionIdx: int): double
+getAvgTrue(): double[]
```

Implement Survey

- Survey objects are intended to be immutable, however…
  - Problem 1: The private field answers can be changed through the reference passed to the constructor.
  - Problem 2: The private field answers can be changed through the reference returned by getAnswers.
- Solution:
  - Copy the boolean array in the constructor before assigning it to answers
  - Copy the private field answers in the method getAnswers before returning the array
- Why is copying done in Survey class instead of Study class?
  - It’s the job of Survey (not Study) to protect the data

iClicker Question

Suppose we change the data type of the answers field (in the class Survey) from boolean[] to ArrayList<Boolean>.

Do we still need to copy answers in the constructor and getAnswers methods?

- a) Yes
- b) No

Show Memory Diagram

```
Study study = new Study();

boolean[] first = {true, false, true};
boolean[] second = {false, true, true};
boolean[] third = {true, false, true};

study.addSurvey(first);
study.addSurvey(second);
study.addSurvey(third);

// What does the following statement print?
System.out.println(Arrays.toString(study.getAvgTrue()));
```

Observations: New Design

- UML diagram appears more complicated, but the classes themselves are simple
  - Each class has a single well-defined purpose (well encapsulated)
  - Easier to write, debug, and (especially) maintain
- The Driver class can create an unlimited number of studies, each with as many surveys as desired

Summary: Key Skills

- Know what data is accessible to each method
  - Instance methods: instance data, class data, parameters, local variables
  - Class methods: class data, parameters, local variables
    - Instance data is not available to class methods
- Pay close attention to the types
  - The type of data determines what can be done with it
- It is much easier to create many small classes than one large class
  - This usually takes some experience to fully embrace (CS 2334)