Part I. Create a class named `TriviaGameV1` that plays a simple trivia game. The game should have five questions. Each question has a corresponding answer and point value between 1 and 3 based on the difficulty of the question. `TriviaGameV1` will use three arrays. An array of type `String` should be used for the questions. Another array of type `String` should be used to store the answers. An array of type `int` should be used for the point values. All three arrays should be declared to be of size 5.

The index into the three arrays can be used to tie the question, answer, and point value together. For example, the item at index 0 for each array would correspond to question 1, answer 1, and the point value for question 1. Manually hardcode the five questions, answers, and point values in the constructor of `TriviaGameV1`. The five questions and their corresponding answers are shown on page 3. The point values should be set to 1, 2, 2, 3, 1, respectively.

The class should also provide the following two public methods:
- `public boolean askNextQuestion()` – This method takes no argument and returns a boolean. If there are no more questions to ask, it returns false. Otherwise, it asks the next question and gets an answer from the user. If the player's answer matches the actual answer (case insensitive comparison), the player wins the number of points for that question. If the player's answer is incorrect, the player wins no points for the question and the method will show the correct answer. It returns true after Q & A have been processed.

- `public void showScore()` – This method takes no argument and returns void. It displays the current score the player receives thus far.

The test driver is provided below, which creates a `TriviaGameV1` object. After the player has answered all five questions, the game is over.

```
public class TriviaGameV1Test
{
   public static void main(String[] args)
   {
      TriviaGameV1 game = new TriviaGameV1();
      while (game.askNextQuestion())
         game.showScore();
      System.out.println("Game over! Thanks for playing!");
   }
}
```
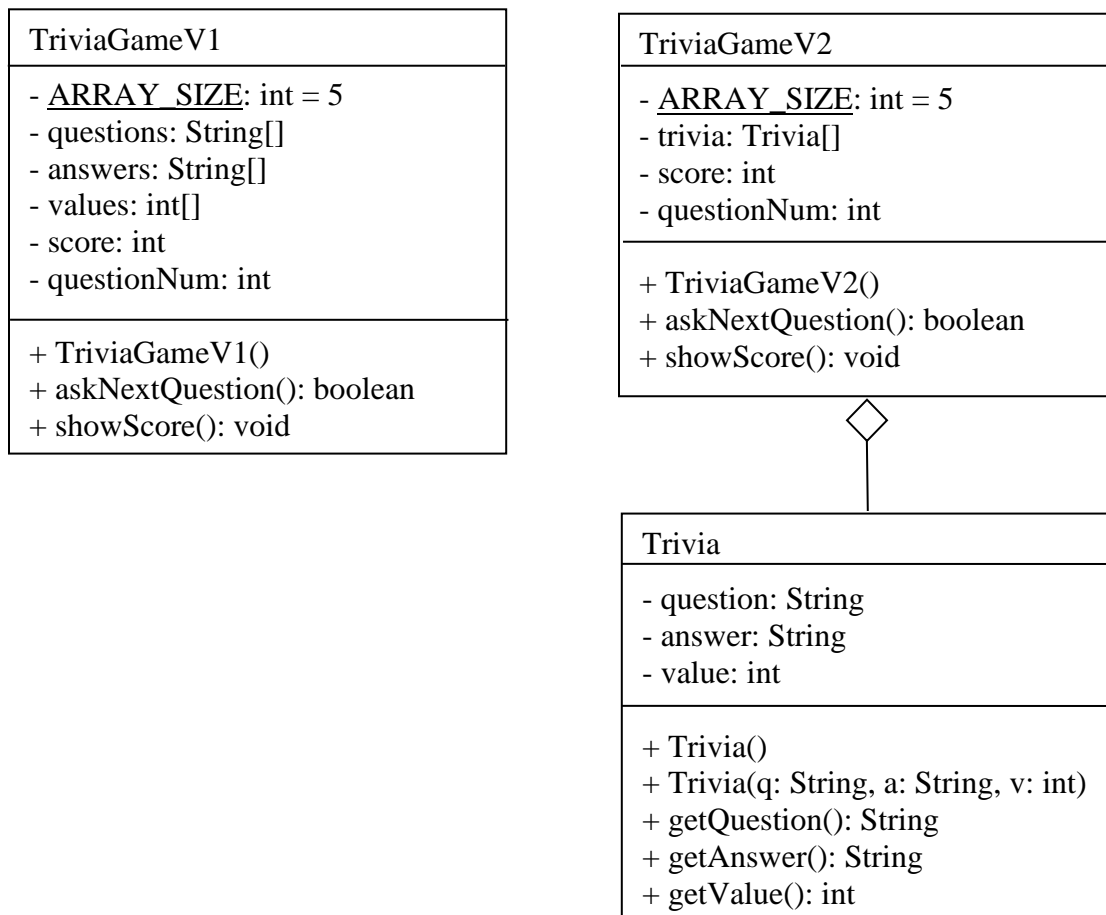
Part II. Write another program that plays the same trivia game. This program will use a single array instead of three arrays. This can be accomplished by first creating a `Trivia` class that encapsulates the question, answer, and point value for a particular trivia question. Next, create a class named `TriviaGameV2` that create a single array of five `Trivia` objects instead of three separate arrays for the question, answer, and point values. This change will make your game more scalable if there were ever additional properties to add to a `Trivia` object. Although the

1

program has internally changed to a single array of objects, the execution of the program should be identical to before. Here is the test driver for `TriviaGameV2`:

```java
public class TriviaGameV2Test
{
    public static void main(String[] args)
    {
        TriviaGameV2 game = new TriviaGameV2();
        while (game.askNextQuestion())
            game.showScore();
        System.out.println("Game over! Thanks for playing!");
    }
}
```

The following are UML diagrams for `TriviaGameV1`, `Trivia`, and `TriviaGameV2`.

The constructor of `TriviaGameV1` should set both `score` and `questionNum` to 0, create three arrays, and manually hardcode questions, answers, and values. The constructor of `TriviaGameV2` is analogous to that of `TriviaGameV1` except that we are handling a single array of five `Trivia` objects.

TriviaGameV1

- ARRAY_SIZE: int = 5
- questions: String[]
- answers: String[]
- values: int[]
- score: int
- questionNum: int

+ TriviaGameV1()
+ askNextQuestion(): boolean
+ showScore(): void

TriviaGameV2

- ARRAY_SIZE: int = 5
- trivia: Trivia[]
- score: int
- questionNum: int

+ TriviaGameV2()
+ askNextQuestion(): boolean
+ showScore(): void

Trivia

- question: String
- answer: String
- value: int

+ Trivia()
+ Trivia(q: String, a: String, v: int)
+ getQuestion(): String
+ getAnswer(): String
+ getValue(): int

The following is a sample output when running the program, where user's input is shown in bold:

```
fcsang@garrison ~/cs1400/project $ java TriviaGameV1Test

Question 1
The first Pokemon that Ash receives from Professor Oak
pikachu
That is correct!
Your score is 1

Question 2
Erling Kagge skiied into here alone on January 7, 1993
south pole
That is correct!
Your score is 3

Question 3
1997 British band that produced 'Tub Thumper'
don't know
Wrong.  The correct answer is chumbawumba
Your score is 3

Question 4
Who is the tallest person on record (8 ft. 11 in) that has lived?
robert wadlow
That is correct!
Your score is 6

Question 5
PT Barnum said "This way to the _____" to attract people to the exit.
egress
That is correct!
Your score is 7
Game over! Thanks for playing!
```

In your `cs1400/project` directory, create five programs named `TriviaGameV1.java`, `TriviaGameV1Test.java`, `Trivia.java`, `TriviaGameV2.java`, and `TriviaGameV2Test.java`. Note that the two driver programs `TriviaGameV1Test.java` and `TriviaGameV2Test.java` have been provided. Your Java programs must begin with the comments below and follow the naming and coding conventions posted on Blackboard.

```
// your name
// CS1400, section 03
// Project 6 – Trivia Game
// date
```

Generate a `script` file `pj6.txt` with appropriate time stamps and the following steps visible:

1. a `pwd` to show the current working directory
2. an `ls -l` to show in long format the files in your `cs1400/project` directory
3. display `TriviaGameV1.java` and `TriviaGameV1Test.java`
4. compile `TriviaGameV1Test.java`
5. run `TriviaGameV1Test` (run your program three times with (a) all five answers correct, (b) all five answers wrong, and (c) mix and match right and wrong)
6. display `Trivia.java`, `TriviaGameV2.java`, and `TriviaGameV2Test.java`
7. compile `TriviaGameV2Test.java`
8. run `TriviaGameV2Test` (run your program three times with (a) all five answers correct, (b) all five answers wrong, and (c) mix and match right and wrong)

Submit `pj6.txt` on Gradescope.