Arrays of Objects

We can make arrays of objects. It is important to know that an array of a class type only stores a reference to an object. It doesn't store a copy of the object!

Here is a simple example with a trivial class:

```
class Num
{
      public int val; // Left as public for simplicity
      public Num()
      {
            val = 0;
      }
      public Num(int n)
      {
            val = n;
      }
}
```

Here is some code in a main that creates an array of 4 Num objects:

```
      Num[] numArray = new Num[4];
```

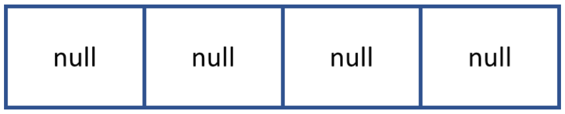
While this makes an array of 4 elements, each element is an empty reference! There is no Num object for us to access. If we print out each object in the array this is what we get:

```
      for (int i = 0; i < numArray.length; i++)
            System.out.println(numArray[i]);
```

Output:

```
      null
      null
      null
      null
```

Null basically means empty, or nothing. We have an array of four nothings!



If we try to access an entry in the array like it's a Num object we'll get an error, because we are trying to access a null, or empty, element.

```
      int v = numArray[0].val;   // Error!
```

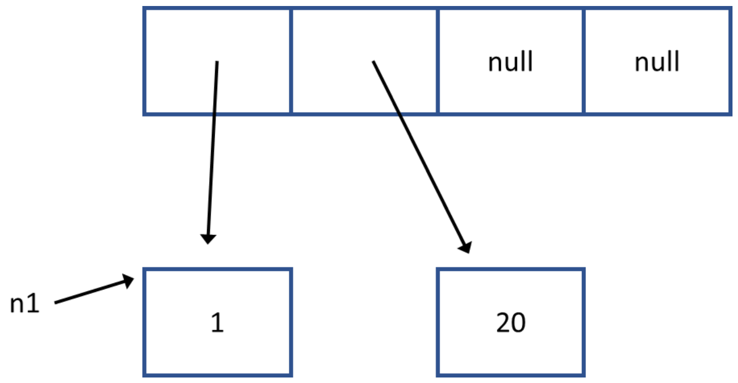
Before we access an array object we have to create it with new. The following shows two ways to set the first two elements, one by referencing a named object, and the other by creating an object and setting it directly in the array.

```
Num[] numArray = new Num[4];

Num n1 = new Num(1);
numArray[0] = n1;
numArray[1] = new Num(20);

for (int i = 0; i < numArray.length; i++)
{
      if (numArray[i]!=null)
            System.out.println(numArray[i].val);
}
// Prints 1 and 20
```

We created objects as shown in the diagram below:



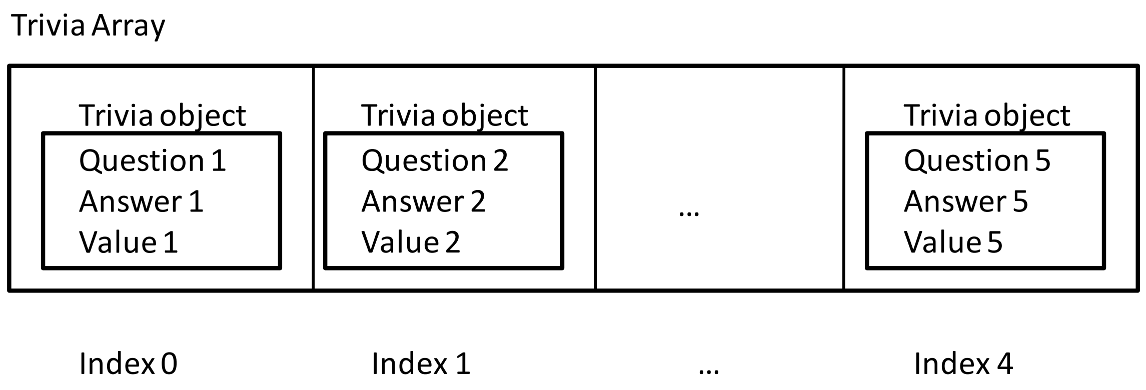
Note that if we change n1.val then the same value would be accessed through numArray[0].val.

The bottom line is to remember that an array of an object type only stores references to instances of objects created by new. If we don't create the individual objects, the array stores null values.

Trivia Game with Classes

As an example of arrays with classes, we can modify our trivia game program to use a single array of a class instead of three separate arrays. The approach with separate arrays is not really scalable if we had more properties associated with each question. For example, if we also wanted to associate an author, date added, and alternate answer, then we would have to add three more arrays. This quickly becomes untenable for a large number of properties. Additionally, if the arrays ever get out of sync (e.g. the answer to question 3 somehow ends up in index 4 instead of index 2) then great havoc will ensue.

A better approach is to make a single object that stores all of the data of interest regarding a trivia question, then make one array of those objects. A diagram is shown below:



First let's make a Trivia class that encapsulates information about one trivia question:

```
public class Trivia
{
      private String question;
      private String answer;
      private int value;

      // Constructor to set the question, answer, and value
      public Trivia(String question, String answer, int value)
      {
            this.question = question;
            this.answer = answer;
            this.value = value;
      }

      // Accessor methods to retrieve the question, answer, value
      public String getQuestion()
      {
            return question;
      }

      public String getAnswer()
      {
            return answer;
      }

      public int getValue()
      {
            return value;
      }
}
```

Now we can modify our TriviaGame class to use the Trivia class instead. We eliminate the three arrays of questions, answers, and values and replace it with a single array called trivia:

```
      private Trivia[] trivia;
```

Our main method now looks like this to set up all the questions, answers, and values:

```
      trivia = new Trivia[NUMQUESTIONS];

      // Manually copy in questions, answers, and values
      // Note that we need to use NEW to create the instance of each
      // object to store in the array

      trivia[0] = new Trivia("The first Pokemon that Ash receives from
Professor Oak", "pikachu",1);

      trivia[1] = new Trivia("This Late Night Talk Show Host studied CS
at Albany College", "Jimmy Fallon", 2);

      trivia[2] = new Trivia("This supermodel has a koding camp for
girls", "Karlie Kloss", 2);

      trivia[3] = new Trivia("A mathematician's lost parrot","polygon",
3);

      trivia[4] = new Trivia("PT Barnum said 'This way to the _______'
to attract people to the exit.", "egress", 1);
```

Note that we use the keyword **new** twice. Once to create the array and then again for each object we place in the array. The creation of the array does not create all the objects that will be stored in the array, that requires another invocation of new.

Here is the rest of the program which is modified to use the trivia array. The only changes needed are in the askNextQuestion method, which must use:

```
      trivia[index].getQuestion(),
      trivia[index].getAnswer(),
      trivia[index].getValue(),
```

in place of:

`questions[index]`   `answers[index]`   and   `value[index]`

```
      // Ask each question
       Scanner keyboard = new Scanner(System.in);
       for (questionNum = 0; questionNum < trivia.length;
           questionNum++)
       {
            System.out.println("Question " + (questionNum+1) + ": " +
                  trivia[questionNum].getQuestion());
            System.out.println("Your answer?");
            String answer = keyboard.nextLine();
            // Check if answer is correct; if so, add to the score
            if
(answer.equalsIgnoreCase(trivia[questionNum].getAnswer()))
            {
                  score+=trivia[questionNum].getValue();
                  System.out.println("That's right! Your new score is "
                      + score);
            }
            else
            {
                  System.out.println("Sorry, the correct answer is " +
                              trivia[questionNum].getAnswer());
            }
       }
       // Output the final score
       System.out.println("The game is over! Your final score is " +
                          score);
      }
```

Question: What would be good method or methods to make this code more readable or more modular?

Later we will see how to read the trivia questions in from a file, which will make this program much more flexible and expandable.