Arrays

March 6, 2012

CMPSCI 121, Spring 2012

Introduction to Problem Solving with Computers

Prof. Learned-Miller

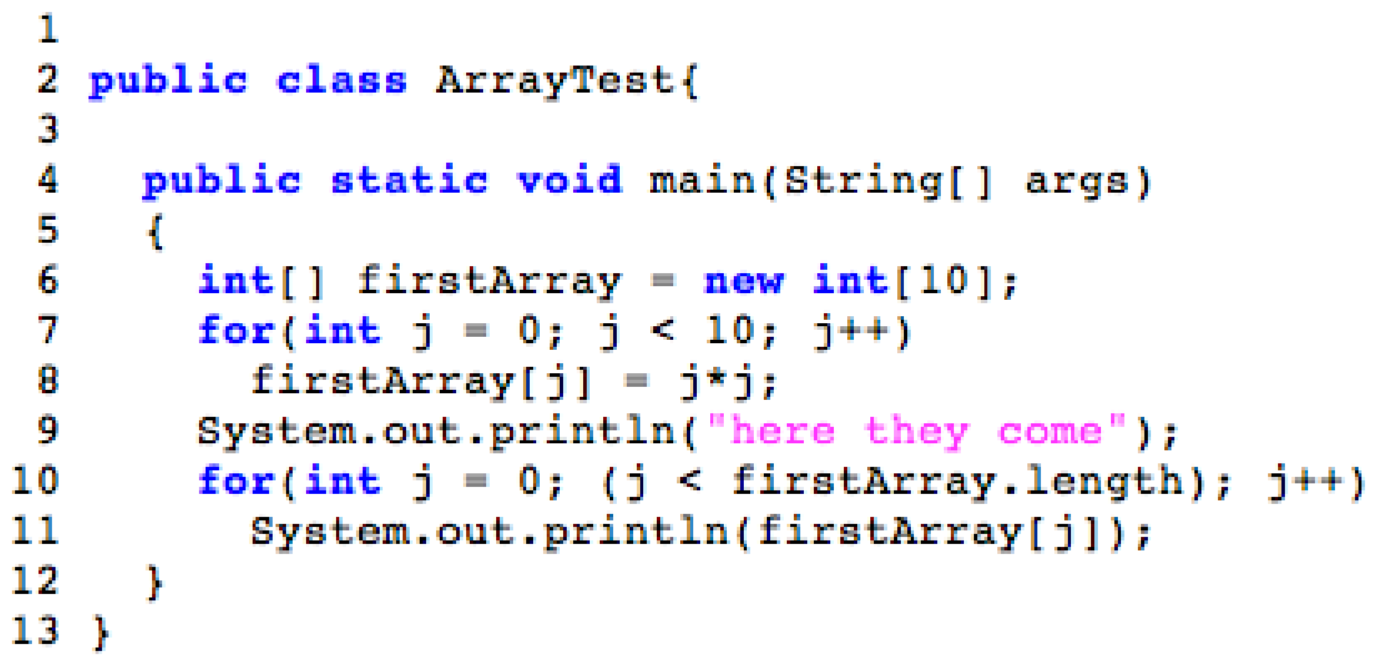
Logistics

- Midterm:
  - Main midterm is Wednesday (starts at 7:30, but can get in at 7:15). Check course web site for details.
- Make-up midterms:
  - Had one last night.
  - Another one tonight: 7:30 pm, Computer Science building, room 142.

Arrays

- For managing multiple instances at a time:
  - Seven ints
  - 275 doubles
  - Three Infants
  - Twelve Integers
  - 17 Strings

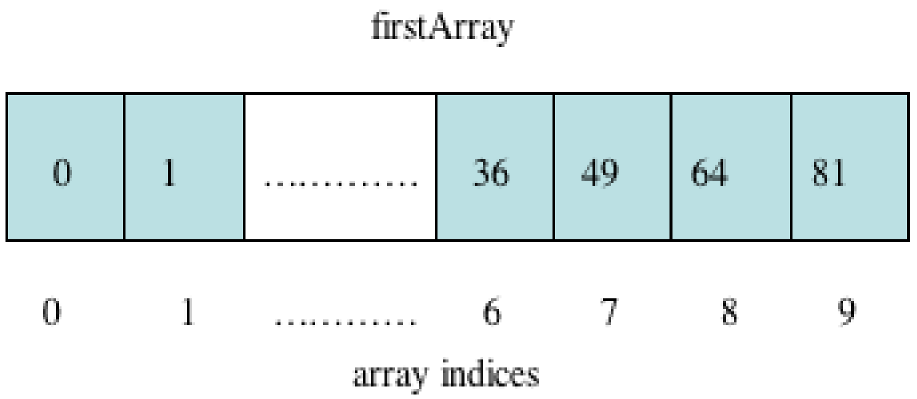
Example



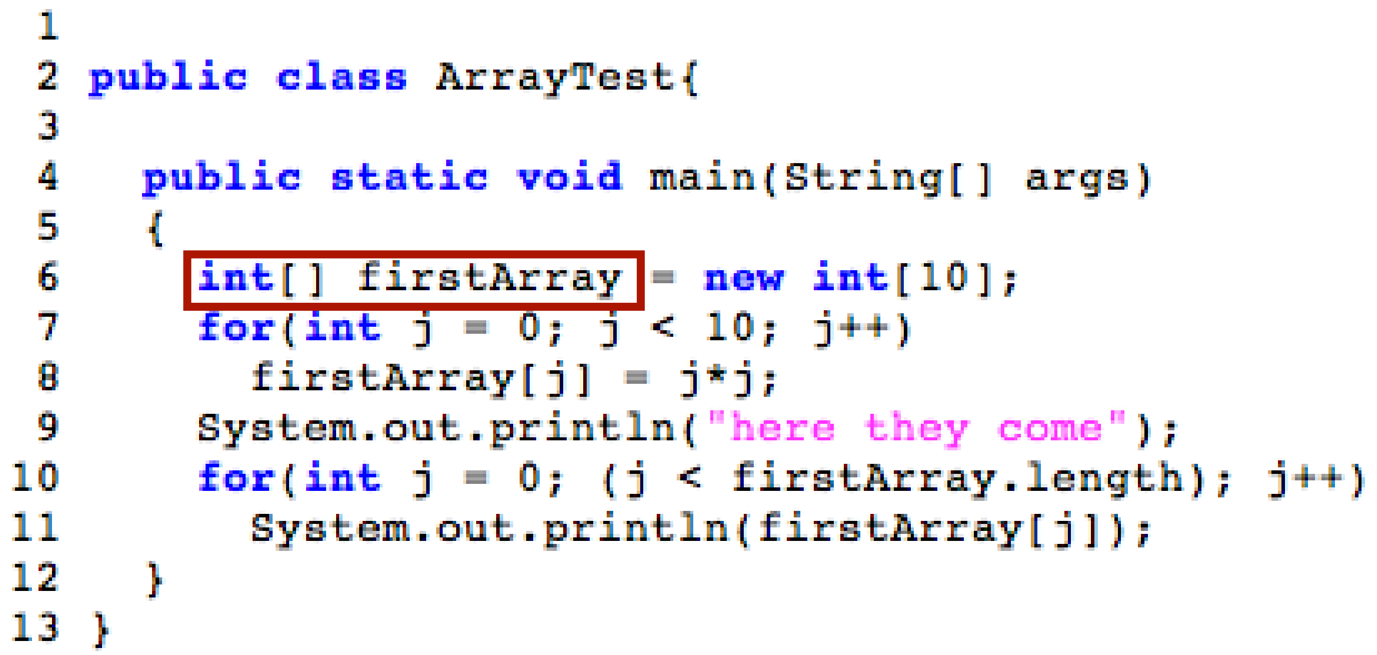
```java

public class ArrayTest{

  public static void main(String[] args)
  {
    int[] firstArray = new int[10];
    for(int j = 0; j < 10; j++)
      firstArray[j] = j*j;
    System.out.println("here they come");
    for(int j = 0; (j < firstArray.length); j++)
      System.out.println(firstArray[j]);
  }
}
```

Example

```

public class ArrayTest{

  public static void main(String[] args)
  {
    int[] firstArray = new int[10];
    for(int j = 0; j < 10; j++)
      firstArray[j] = j*j;
    System.out.println("here they come");
    for(int j = 0; (j < firstArray.length); j++)
      System.out.println(firstArray[j]);
  }
}
```

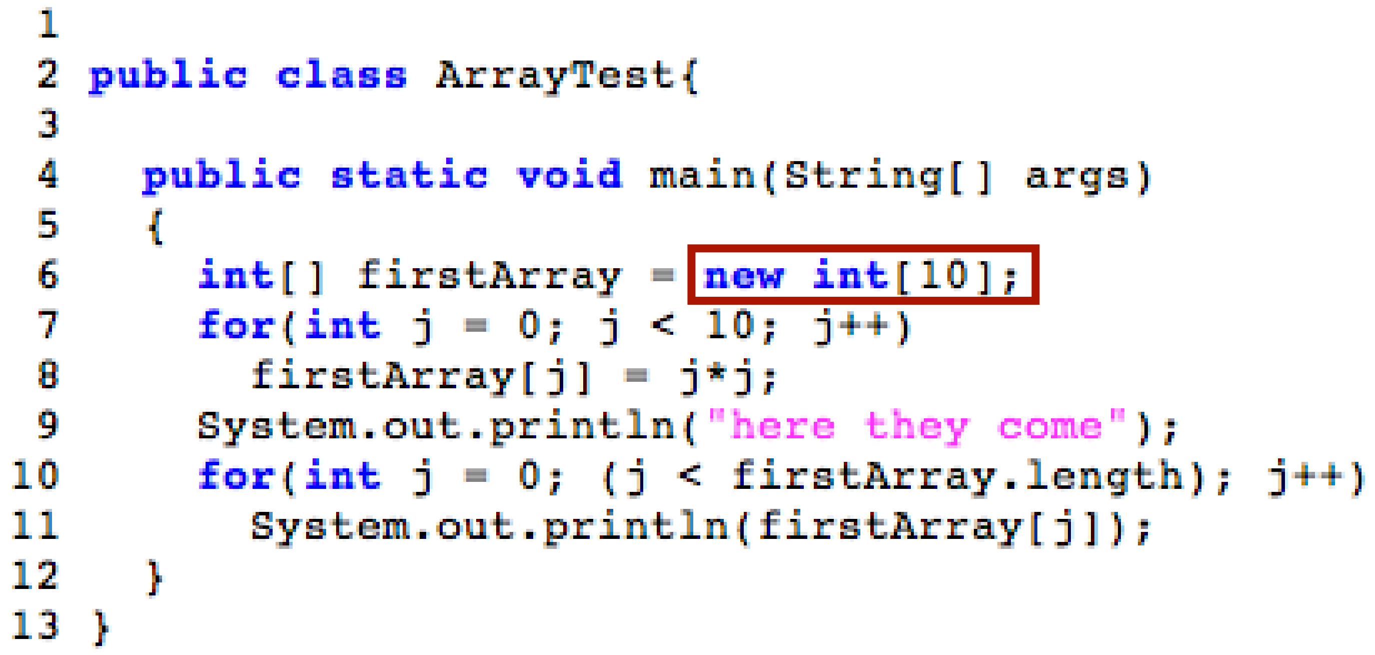


Example



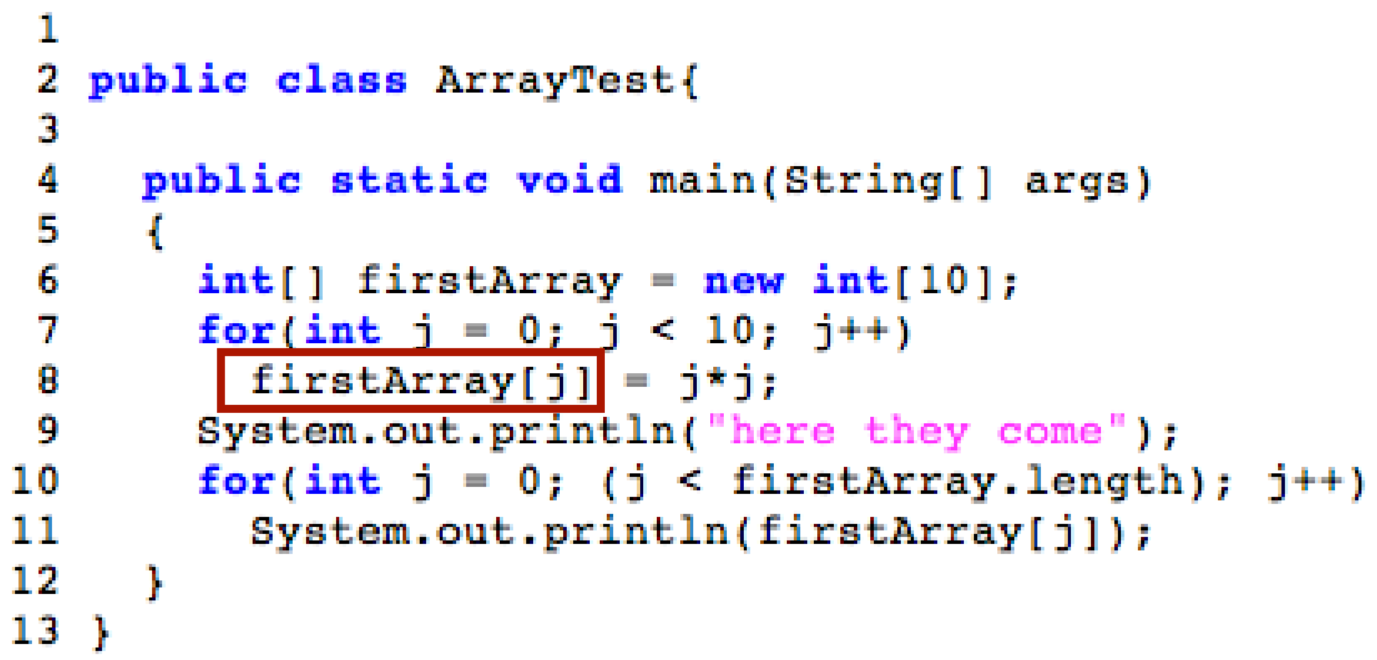
```java

public class ArrayTest{

  public static void main(String[] args)
  {
    int[] firstArray = new int[10];
    for(int j = 0; j < 10; j++)
      firstArray[j] = j*j;
    System.out.println("here they come");
    for(int j = 0; (j < firstArray.length); j++)
      System.out.println(firstArray[j]);
  }
}
```

Example



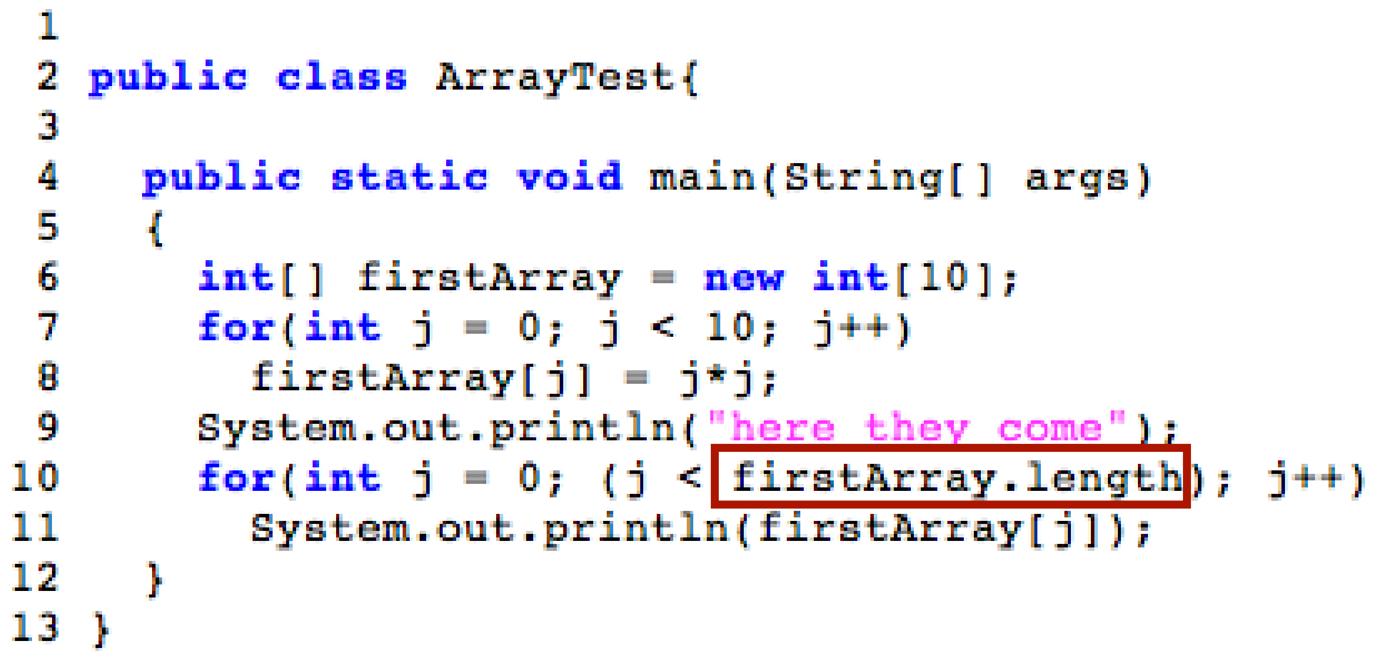
```
public class ArrayTest{

  public static void main(String[] args)
  {
    int[] firstArray = new int[10];
    for(int j = 0; j < 10; j++)
      firstArray[j] = j*j;
    System.out.println("here they come");
    for(int j = 0; (j < firstArray.length); j++)
      System.out.println(firstArray[j]);
  }
}
```

Example



```java

public class ArrayTest{

  public static void main(String[] args)
  {
    int[] firstArray = new int[10];
    for(int j = 0; j < 10; j++)
      firstArray[j] = j*j;
    System.out.println("here they come");
    for(int j = 0; (j < firstArray.length); j++)
      System.out.println(firstArray[j]);
  }
}
```

Example



```java

public class ArrayTest{

  public static void main(String[] args)
  {
    int[] firstArray = new int[10];
    for(int j = 0; j < 10; j++)
      firstArray[j] = j*j;
    System.out.println("here they come");
    for(int j = 0; (j < firstArray.length); j++)
      System.out.println(firstArray[j]);
  }
}
```

Accessing Elements of Arrays

- For an array of ints called foo
  - foo[3] is the fourth int.

Accessing Elements of Arrays

- For an array of doubles called blech
  - blech[7] is the eighth double.

Accessing Elements of Arrays

- For an array of Infants called kindergarten
  - kindergarten[12] is the thirteenth Infant.

Get ‘em or Set ‘em

- int x = foo[3];
- foo[6] = 12;
- foo[17] = foo[2];

Get ‘em or Set ‘em

- foo[5]+=14;
- foo[5] *= foo[2];
- foo[x+3] = 3;
- foo[foo[3]] = 2;

// Treat “foo[3]” just like any other variable, like “x”.

Declaring Arrays

- int[ ] foo;   // foo is array of ints.
- double[ ] blech;   // array of doubles.
- Infant[ ] kindergarten;   // array of Infants

Initializing and Allocating Arrays

Assume foo has been declared:

- int [ ] foo;

Now initialize to an array:

- foo = new int [27];

Declaration and Initialization:

- int [ ] foo = new int [27];

Assigning to Arrays

Assume we have declared an array of ints

- int [ ] foo;

Two parts to the assignment!!!

Set up the “boxes”

- foo = new int [3];

Fill the boxes.

- foo[0]=7;
- foo[1]=12;
- foo[2]=3;

Parking Lot Analogy

- int [ ] foo; → The name of the parking lot will be "foo" and it will hold "ints".
- foo = new int[3]; → Paint lines for 3 ints.

Assignments: → Place ints in the 3 spots.

- foo[0] = 7;
- foo[1] = 12;
- foo[2] = -2;

Parking Lot Analogy

- Car [ ] foo; $\longrightarrow$ The name of the parking lot will be “foo” and it will hold “Cars”.
- foo = new Car[4]; $\longrightarrow$ Paint lines for 4 Cars.
- Assignments: $\longrightarrow$ Place Cars in the 4 spots.
  - foo[0] = new Car(“BMW”);
  - foo[1] = new Car(“Audi”);
  - foo[2] = new Car(“VW”);
  - foo[3] = new Car(“Yugo”);

Parking Lot Analogy

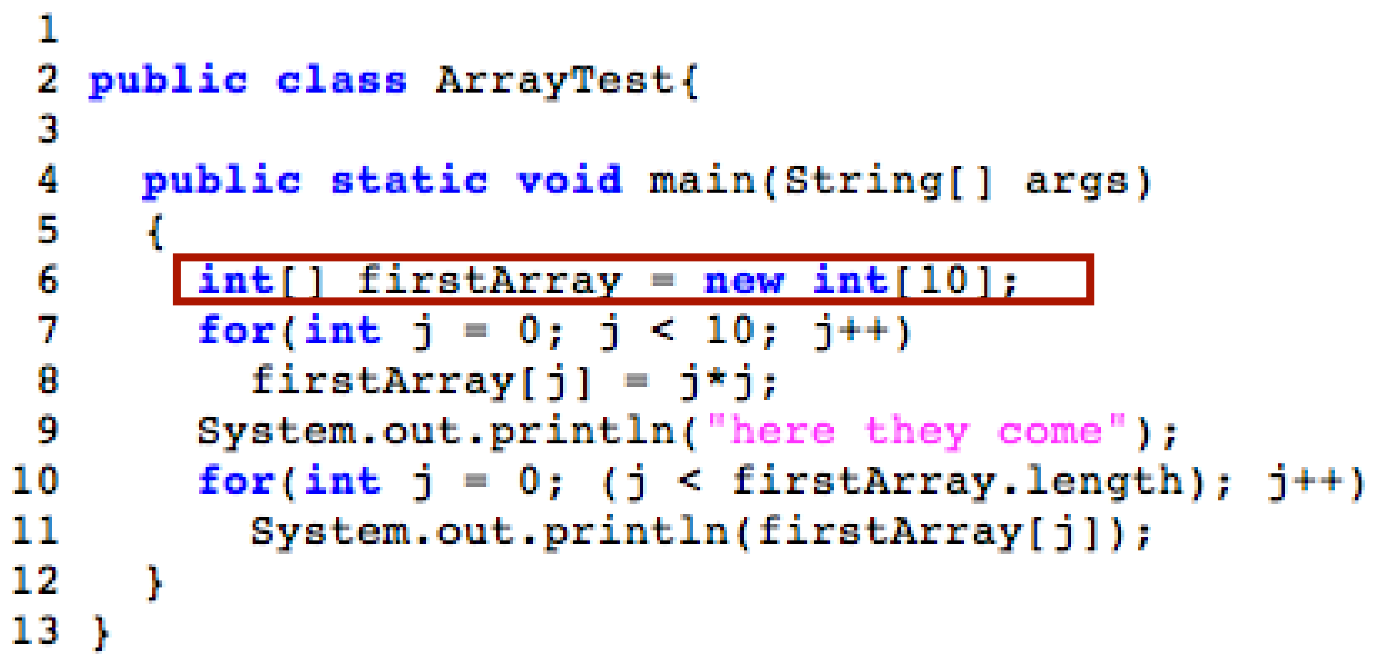
- Car [ ] foo; $\longrightarrow$
- foo = new Car[4]; $\longrightarrow$
- Assignments: $\longrightarrow$
  - foo[0] = $\boxed{\text{new Car(“BMW”);}}$
  - foo[1] = new Car(“Audi”);
  - foo[2] = new Car(“VW”);
  - foo[3] = new Car(“Yugo”);

- The name of the parking lot will be “foo” and it will hold “Cars”.
- Paint lines for 4 Cars.
- Make 4 cars and place them in the 4 spots.

Summary

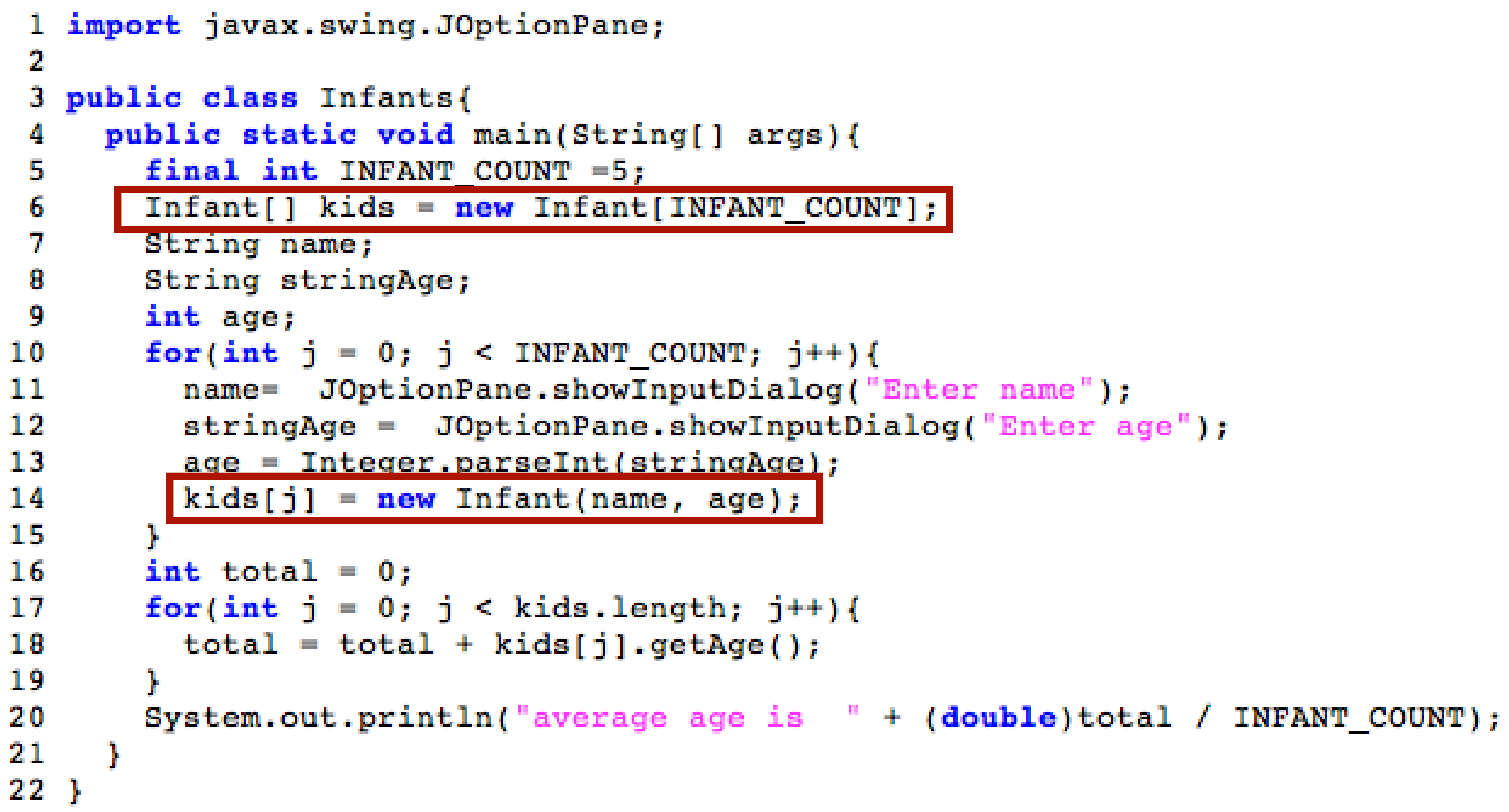
- For primitive types:
  - Call “new” once to create "spaces" for all elements of array.
- For Objects:
  - Call “new” twice:
    - once to create “spaces” for object references
    - once to create each object.

Array of Primitive Types



```java

public class ArrayTest{

  public static void main(String[] args)
  {
    int[] firstArray = new int[10];
    for(int j = 0; j < 10; j++)
      firstArray[j] = j*j;
    System.out.println("here they come");
    for(int j = 0; (j < firstArray.length); j++)
      System.out.println(firstArray[j]);
  }
}
```

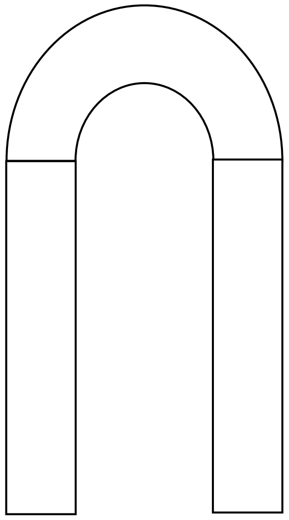
Array of Objects



```java
import javax.swing.JOptionPane;

public class Infants{
  public static void main(String[] args){
    final int INFANT_COUNT =5;
    Infant[] kids = new Infant[INFANT_COUNT];
    String name;
    String stringAge;
    int age;
    for(int j = 0; j < INFANT_COUNT; j++){
      name=  JOptionPane.showInputDialog("Enter name");
      stringAge =  JOptionPane.showInputDialog("Enter age");
      age = Integer.parseInt(stringAge);
      kids[j] = new Infant(name, age);
    }
    int total = 0;
    for(int j = 0; j < kids.length; j++){
      total = total + kids[j].getAge();
    }
    System.out.println("average age is  " + (double)total / INFANT_COUNT);
  }
}
```

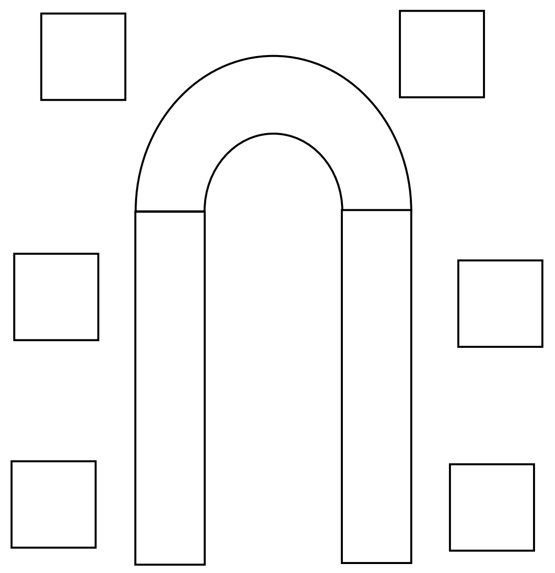
Graphical Illustrations of Array Allocation

```
// Declaration: “Where” is it?
House [ ]  neighborhood;
```



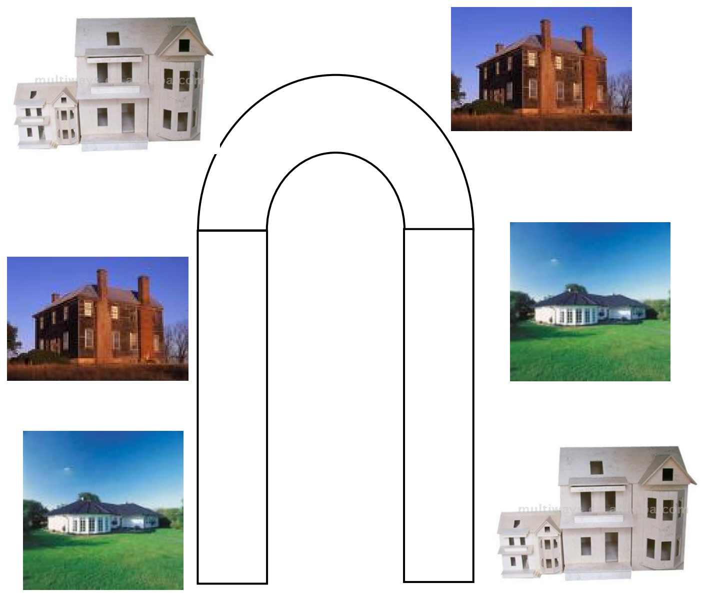
Graphical Illustrations of Array Allocation

```
// Array allocation: “How many” lots?
neighborhood = new House[6];
```



Graphical Illustrations of Array Allocation

```
// Initialization of Objects: What do they look like?
for (int i=0; i<6; i++)
    neighborhood[i]=new House(style,squareFeet);
```



Memory and Arrays

- How big can you make your arrays?
- Computer memory (RAM)
  - Typical size: 1 Gigabyte = 1,000,000,000 bytes
- ints: 4 bytes apiece
  - 1,000,000,000/4 = 250 Million
- doubles: 8 bytes apiece
  - 1,000,000,000/8 = 125 Million
- Infants:
  - memory address at beginning = 4 bytes
  - name average 10 chars = 20 bytes
  - age stored as int = 4 bytes
  - 1,000,000,000/28 = 35.7 million

DrJava