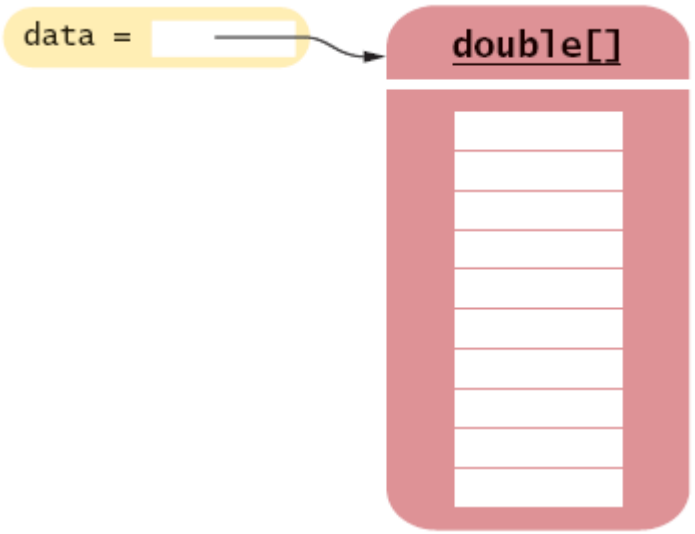
CompSci 6
Programming Design and Analysis



February 2, 2010

Prof. Rodger

Announcements

- Read for next time Chap. 6.3-6.5
  - More on loops, randomness
- Reading Quiz for next time
- Today – Loops and Arrays
- Classwork
  - APTs with arrays and loops

Both while and for loops

- Initialization
- Condition
- Body
- Increment

while

```
public void printFencePost(int numberPosts) {
    String rail = "===";
    String post = "I";

    int num = 1;
    System.out.print(post);
    while (num < numberPosts) {
        System.out.print(rail);
        System.out.print(post);
        num++;
    }
    System.out.println(" ");
}
```

while (cont)

```
x.printFencePost(6);
x.printFencePost(5);
x.printFencePost(12);
```

```
I===I===I===I===I===I
I===I===I===I===I
I===I===I===I===I===I===I===I===I===I===I===I===I
```

for loop

```
public void printFencePostfor(int numberPosts) {
    String rail = "===";
    String post = "I";

    System.out.print(post);
    for (int k = 1; k < numberPosts; k++) {
        System.out.print(rail);
        System.out.print(post);
    }
    System.out.println(" ");
}
```

Arrays

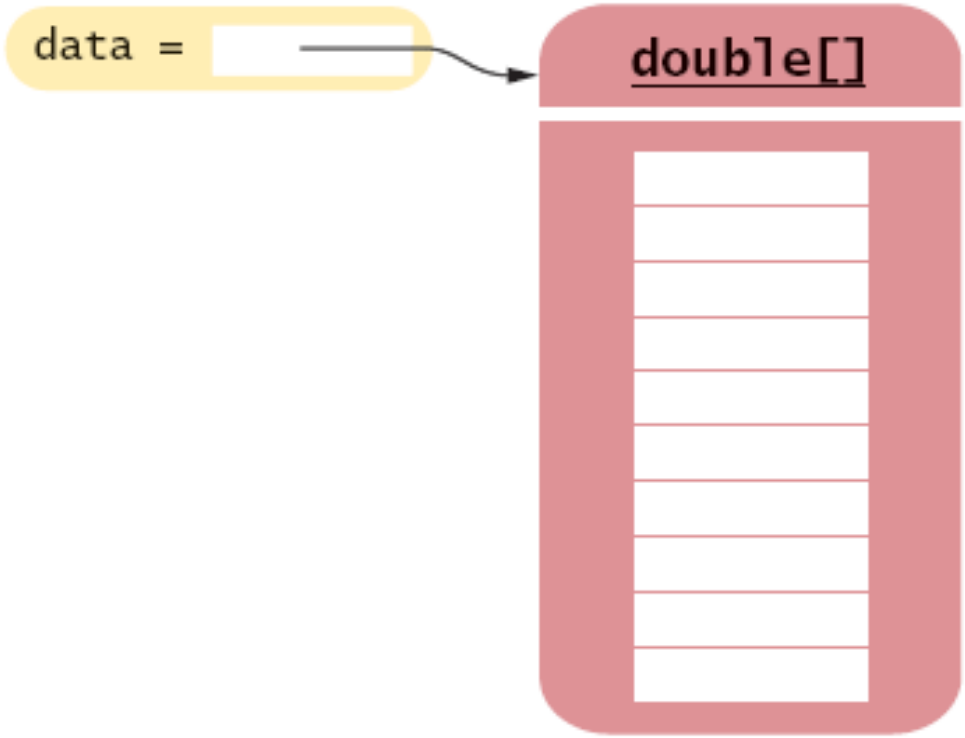


Figure 1 An Array Reference and an Array

Array Access

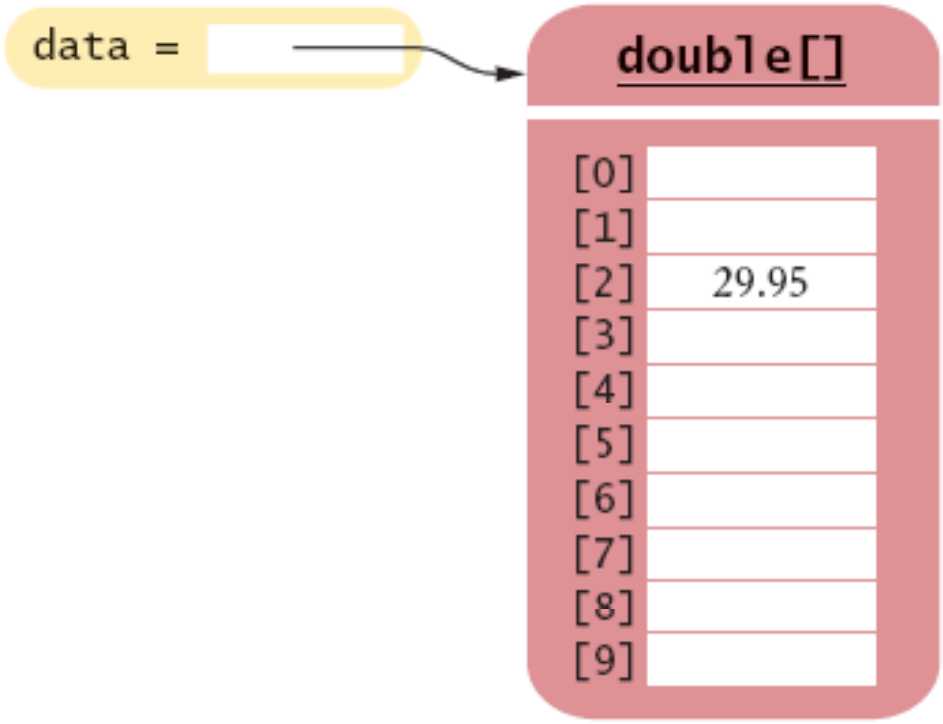


Figure 2 Storing a Value in an Array

Array Syntax

- Creating an array

  new *typeName*[*length*]

  Example: `new double[10]`

  Purpose: To construct an array with a given number of elements.

- Accessing elements

  arrayReference[*index*]

  Example: data[2]

  Purpose: To access an element in an array.

Array

- Declare and initialize an array of integers

```
int[] values = new int[12];
```

- Set it to these values:

```
8 3 4 3 8 2 4 4 6 2 8 4
```

- Access item in slot 6 in the array

```
values[6]
```

- Array is fixed size. The size is:

```
values.length
```

Self Check 7.1

What elements does the data array contain after the following statements?

```
double[] data = new double[10];
for (int i = 0; i < data.length; i++)
{
    data[i] = i * i;
}
```

Answer:

Self Check 7.2

What do the following program segments print? Or, if there is an error, describe the error and specify whether it is detected at compile-time or at run-time.

a)
```
double[] a = new double[10];
System.out.println(a[0]);
```

b)
```
double[] b = new double[10];
System.out.println(b[10]);
```

c)
```
double[] c;
System.out.println(c[0]);
```

Answer:

a)

b)

c)

Loops

- Traverses all elements of a collection:

```
double[] data = ...;
   double sum = 0;
   for (double e : data)
      // Read this loop as
      // "for each e in data"
   {
      sum = sum + e;
   }
```

- Traditional alternative:

```
double[] data = ...;
   double sum = 0;
   for (int i = 0; i <
   data.length; i++)
   {
      double e = data[i];
      sum = sum + e;
   }
```

ArrayList

- Class vs. primitive
- ArrayList
  - Can grow and shrink
  - Has methods for common tasks (see API)
  - Only holds objects
- Can’t have an ArrayList of int or double
  - Need to use wrapper class like Integer or Double

ArrayList (cont)

- Create an ArrayList

```
ArrayList<Integer> idlist = new ArrayList<Integer>();
```

- Add an element to the ArrayList

```
idlist.add(8);
```

- Modify kth element in an ArrayList

```
idlist.set(k,8);
```

- Sum the elements in the ArravList

```
// sum up integers in the ArrayList
int sum = 0;
for (Integer current : idlist)
{
    sum += current;
}
```

ArrayList vs. array

- Methods
  - Sort an arrayList called numbers

    ```
    Collections.sort(numbers);
    ```

  - Sort an array called a

    ```
    Arrays.sort(a);
    ```

- Types
  - Arrays can hold any type
  - ArrayLists only work with objects
- ArrayList's are dynamic – easy to expand in size
- Can convert from one to the other
- APTs only pass and return arrays

Example: singleNumbers

- Given an integer array that could have duplicates, return an array that has only unique numbers from the original array (get rid of duplicates!)
- For example if the parameter array is:
  - 8 5 5 8 5
- Then the array to return should be:
  - 8 5

First convert array to ArrayList

```
public int[] singleNumbers(int[] ids) {

    // convert the array "ids" into an ArrayList "idlist"
    ArrayList<Integer> idlist = new ArrayList<Integer>();
    for (int k = 0; k < ids.length; k++) {
        idlist.add(ids[k]);
    }
```

Second, find unique numbers

```
// create an ArrayList that will hold unique numbers
ArrayList<Integer> singles = new ArrayList<Integer>();
singles.add(idlist.get(0)); // first number is unique
for (Integer current : idlist) {
    boolean isIn = false;
    for (Integer currentSingle : singles) {
        if (current.equals(currentSingle))
            isIn = true;
    }
    if (!isIn)
        singles.add(current);
}
```

Third, convert ArrayList to Array

```
// convert ArrayList to array
int[] answer = new int[singles.size()];
int position = 0;
for (Integer currentSingle : singles) {
    answer[position] = currentSingle;
    position++;
}

return answer;
```

or…

- Convert ArrayList to array

Use ArrayList's `toArray()` method

```
Integer[] answer =
  (Integer[])singles.toArray();
```

- Convert array to ArrayList

Use Array's static `asList()` method

```
ArrayList<String> nameList =
  (ArrayList<String>)Arrays.asList(names)
  ;
```

  - *Only works with Objects not primitive types*
  - *names is an array of Strings*

Classwork today - APT

- AimToTen
- AccessLevel