Arrays

GEEN163

"Any fool can write code that a computer can understand. Good programmers write code that humans can understand."

Martin Fowler

TuringsCraft

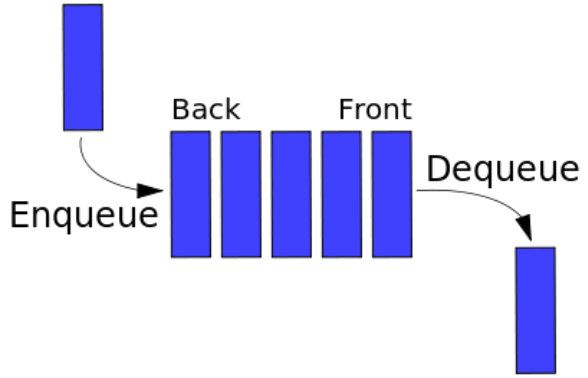
- Read chapter 8 of the textbook on arrays
- Answer questions in section 8 of the TuringsCraft tutorial system
  - 4 points for each correct answer
  - maximum of 100 points
- Due by midnight on **Tuesday**

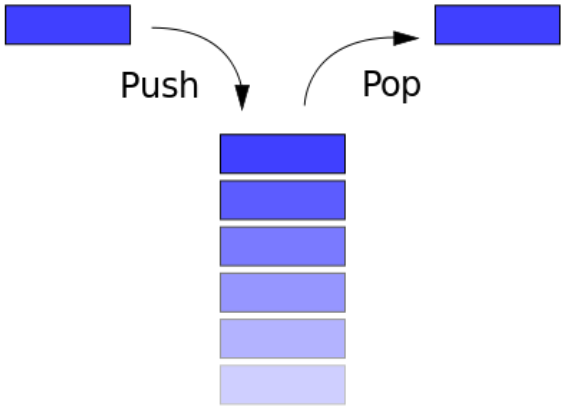
Lots of Data

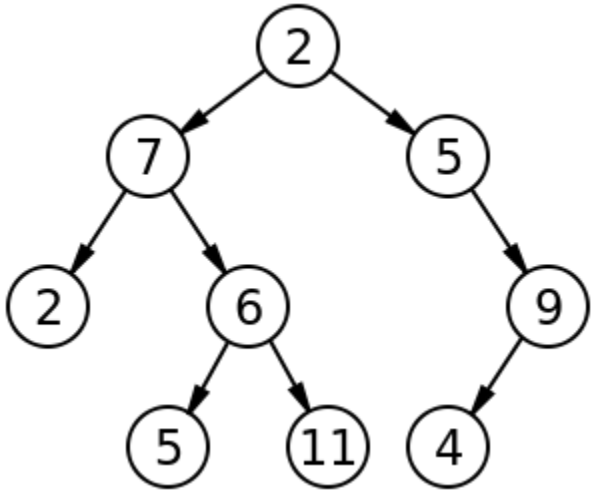
- Sometimes programs need to deal with lots of data
- A program that manages a list of the students in a course should not have to create separate variables for each student
- **Arrays** are a means of holding lots of data in a single variable while still being able to access each individual data item

Data Structures

- Data Structures are ways of organizing data in a program
- Popular data structures are
  - array
  - tree
  - hash table
  - queue
  - stack







Arrays

- An array is a set of memory locations with the same name
- All of the variables are of the same type
- You can use an integer to specify which member of the array you are using
- You can use a member of an array anyplace you would use a regular variable

Vectors in Mathematics

- Mathematicians often use subscripted variables to represent elements in a vector

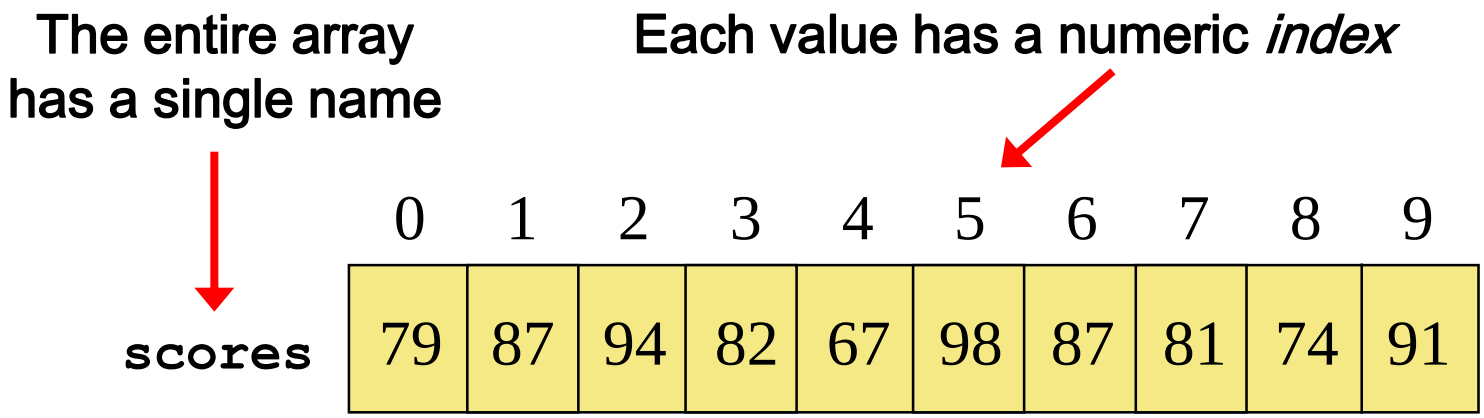
$$a_1, a_2, \ldots a_{12}$$

- Arrays are the computer implementation of a vector
- Indexes are used to specify a member of an array. In Java the index follows the array name in square brackets

```
a[3] = 678.9;
```

Arrays

- An array is an ordered list of values:



An array of size N is indexed from zero to N-1

This array holds 10 values that are indexed from 0 to 9

Memory Boxes

- An array can be thought of as a row of boxes where each box is a memory location that can store a value
- Each box is numbered starting from zero
- You can specify which box you want to use by indicating the number on the box

Arrays

- A particular value in an array is referenced using the array name followed by the index in brackets
- For example, the expression

```
scores[2]
```

refers to the value **94** (the 3rd value in the array)

- That expression represents a place to store a single integer and can be used wherever an integer variable can be used

Array Use

- For example, an array element can be assigned a value, printed, or used in a calculation:

```
scores[2] = 89;

scores[first] = scores[first] + 2;

mean = (scores[0] + scores[1])/2;

System.out.println ("Top = " + scores[5]);

pick = scores[rand.nextInt(10)];
```

When you use an array, the value inside the square brackets is the

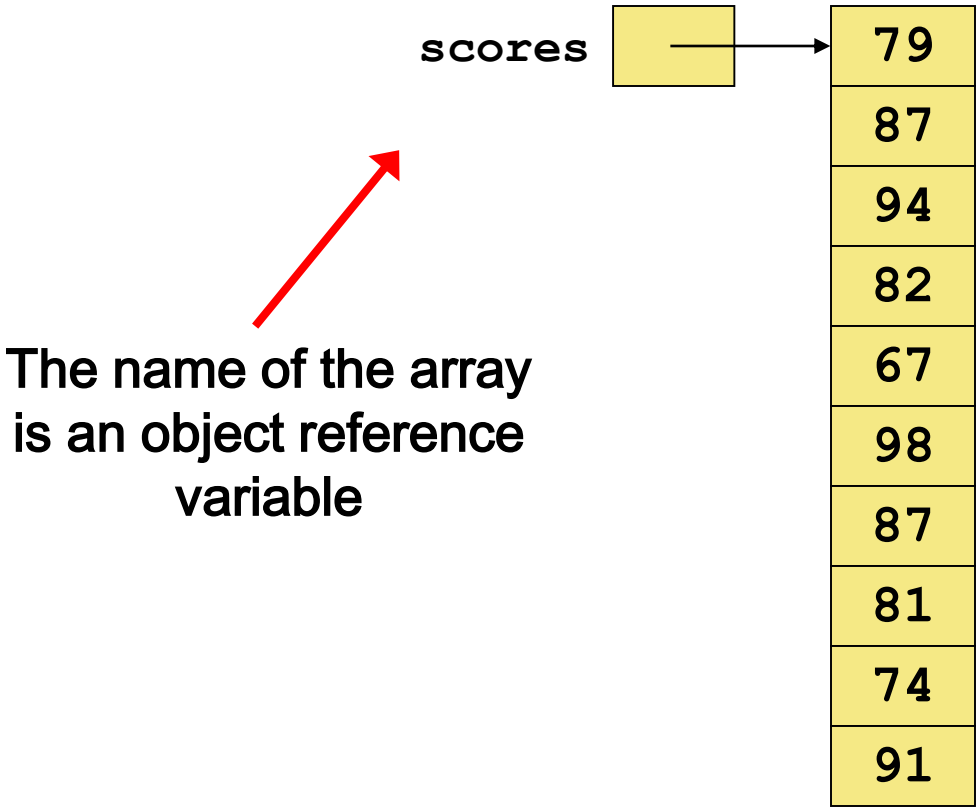
A. element
B. index
C. length
D. array name

Arrays

- The values held in an array are called *array elements*
- An array stores multiple values of the same type – the *element type*
- The element type can be a primitive type or an object reference
- You can create an array of integers, an array of characters, an array of **`String`** objects, an array of Widget objects, etc.

Arrays are Objects

- In Java, the array itself is an object that must be instantiated
- Another way to depict the **`scores`** array:



Declaring Arrays

- The **`scores`** array could be declared as follows:

```
int[] scores = new int[10];
```

- The type of the variable **`scores`** is **`int[]`** (an array of integers)
- Note that the array type does not specify its size, but each object of that type has a specific size
- The reference variable **`scores`** is set to a new array object that can hold 10 integers

Declaring Arrays

- Some other examples of array declarations:

```
int[] weights = new int[2000];

double[] prices = new double[500];

boolean[] flags;
flags = new boolean[20];

char[] codes = new char[1750];
```

Array Example

- The following program reads 17 numbers from the keyboard then prints them in reverse order

```
public class PrtArray {
  public static void main(String[] args){
    java.util.Scanner kb = new
                java.util.Scanner(System.in);
    int[] cat = new int[17];

    for (int bird = 0; bird < 17; bird++) {
       cat[bird] = kb.nextInt();
    }

    for (int dog=16; dog >= 0; dog--) {
       System.out.println(cat[dog]);
    }
  }
}
```

How do you declare an array of 7 ints?

A. `int dog[7] = new int[];`
B. `int dog = new int[7];`
C. `int[] dog = new int[7];`
D. `int[7] dog = new int[7];`
E. `int dog[] = new int[7];`

Bounds Checking

- Once an array is created, it has a fixed size
- An index used in an array reference must specify a value in range 0 to N-1 for an N element array
- The Java system throws an **`ArrayIndexOutOfBoundsException`** if an array index is too big or negative
- This is called automatic *bounds checking*

Bad Index

```
int[]  rabbit = new int[4];
rabbit[0] = 47;
rabbit[3] = 32;
rabbit[4] = 14;    // this gets an error
```

- If an array is created with N elements, you can index from zero to N-1
- Bigger numbers generate an error

Off by One

- For example, if the array **`canary`** can hold 100 values, it can be indexed from 0 to 99
- It's common to introduce ***off-by-one errors*** when using arrays:

```
double[] canary = new double[100];
for(int dove = 0;dove <= 100;dove++){
    canary[dove] = 0.0;
}
```

- This will get a run time error

Use of `for` with Arrays

- The correct way to use a for loop with an array is to start at zero and go to less than the size

```
double[] canary = new double[100];
for(int dove = 0;dove < 100;dove++){
    canary[dove] = 0.0;
}
```

Size of an Array

- Each array object has a public constant called **`length`** that stores the size of the array
- It is referenced using the array name

```
double[] canary = new double[100];

System.out.println(canary.length);
```

- Note that **`length`** holds the number of elements, not the largest index

Using the Length

- You can use the length value to always loop the correct number of times

```
double[] kennel = new double[something];
for (int index = 0; index <  kennel.length; index++) {
        kennel[index] = index;
}
```

What is displayed?

```
int[] rabbit = new int[5];
for (int i = 0; i < 5; i++) {
   rabbit[i] = 2 * i;
}
System.out.println(rabbit[2]);
```

A. 0

B. 2

C. 4

D. 10

What is displayed?

```
int[] rabbit = new int[5];
for (int i = 0; i < 5; i++) {
   rabbit[i] = 2 * i;
}
System.out.println(rabbit[7]);
```

A. 0

B. 2

C. 14

D. none of the above

Things Arrays Do **NOT** Do

Assume x, y and z are arrays

- Add, subtract, multiple or divide arrays

```
x = y * z;
```

- Compare arrays

```
if ( x == y )  // valid syntax!
```

- read or write arrays as one unit

```
System.out.println( x ); // valid syntax!
```

Using Elements

- When you use an array you almost always have to specify an index in brackets

```
int dog;
dog = cat[14];
```

- The only time you do not use brackets is when you pass the entire array to a method or use .length

Which for loop should be used to set all values of deer to 47?

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

A. `for(int i = 0; i < deer; i++){`

B. `for(int i = 0; i <= deer; i++){`

C. `for(int i = 0; i < deer.length; i++){`

D. `for(int i = 0; i <= deer.length; i++){`

```
        deer[i] = 47;
}
```

For Loops and Arrays

- Arrays are almost always used in a loop
- The reason `for` loops often start at zero instead of one it because array indexes start at zero

For Loop for Arrays

- A for loop is frequently used to go through an array

```
double sum = 0.0;
double[] dog = new double[??];
        // put something in dog
for (int i = 0; i < dog.length; i++ ) {
    sum += dog[i];
}
```

Extended for Loop

- There is a special format for the for loop to go through an array

```
double sum = 0.0;
double[] dog = new double[??];
        // put something in dog
for ( double cat : dog ) {
    sum += cat;
}
```

Extended for

- In the extended for statement

for (type element : array)

- in each iteration of the loop, the next member of the array is copied to the element
- The element must be the same type as the elements of the array
- The extended for loop can be used for arrays or **iterable** classes

Some Iterable Classes

AbstractCollection
AbstractList
AbstractQueue
AbstractSequentialList
AbstractSet
ArrayList
ConcurrentLinkedDeque
ConcurrentLinkedQueue
HashSet
LinkedBlockingDeque
LinkedBlockingQueue
LinkedHashSet
LinkedList
LinkedTransferQueue
PriorityBlockingQueue
PriorityQueue
Stack
SynchronousQueue
TreeSet
Vector

Irritable, not Iterable

Command Line

- The parameters to the main method is an array of Strings

  public static void main(String[] args)

- If you execute the compiled program from the command line as

```
java myprog  PIG  cow
```

then

args[0] is “PIG”

args[1] is “cow”

Write a Java segment

Assume that the array cat has been created

```
double[] cat = new double[1000];
```

and filled with numbers.

Write a loop to display the biggest number in the array.

Possible Solution

```
double big = cat[0];
for (int eye = 1; eye < cat.length; eye++) {
     if (cat[eye] > big ) {
          big = cat[eye];
     }
}
System.out.println( big );
```

TuringsCraft

- Read chapter 8 of the textbook on arrays
- Answer questions in section 8 of the TuringsCraft tutorial system
  - 4 points for each correct answer
  - maximum of 100 points
- Due by midnight on **Tuesday**