Arrays

Lecture 20

Borrowing from slides by Alan Hu, Kurt Eiselt, Paul Carter, and Tamara Munzner
News

- Assignment 2
  - Due this Friday, Oct 30

- Midterm 2 coming up next week
  - Wednesday, Nov 4, 6:30-7:30
  - Note: the rooms will be different
  - Material: conditionals, loops
    - Arrays will NOT be on the midterm
Reading Assignments

- Reading for this week: arrays
  - Edition 3: Ch. 7.1, 7.5-7.7
  - Edition 2: Ch. 8.1, 8.5-8.7
Objectives for Today

- Practice loops and conditionals with one more examples
- Learn about the principle of root finding with nested intervals
- Gain an initial understanding of arrays
One More Example: Loan Amortization

- How do we compute the right payment amount to pay off a loan after some number of years?
We Have Seen This Code Before

- How about computing your account balance after some number of years?

```java
double balance = initBalance;
int years = 0;
while (years < term) {
    intAccrue = balance*intRate/100;
    balance += (intAccrue - payment);
    years++;
}
```
Loan Amortization

- How do we compute the right payment amount to pay off a loan after some number of years?

- Use “Nested Intervals” approach:
  - Have two guesses: tooLow and tooHigh
  - Try a guess halfway in between.
  - Compute the loan balance using guess.
  - If guess was too high, then tooHigh = guess else tooLow = guess.
  - Repeat
Loan Amortization

- How do we know that our program will terminate?
- How do we know that our program will compute the correct result?
Loan Amortization

How do we know that our program will terminate?

- Inner Loop: \textit{years} starts at 0 and counts up to \textit{term}. Will always execute exactly \textit{term} times.

- Outer Loop: The gap between \texttt{tooLow} and \texttt{tooHigh} gets cut in half each iteration.

These are termination arguments (aka ranking functions). You should always know why your loops will terminate.
Loan Amortization

- How do we know that our program will compute the correct result?
  - Inner Loop: At each iteration, balance is always the correct value after years years.
  - Outer Loop: At each iteration, tooLow is always less than the correct value, and tooHigh is always greater than the correct value.

- These are called “loop invariants”. Very helpful to understand loops. (Great to put in comments!)
Arrays
Arrays

- Arrays let you create a bunch of variables (all of the same type), that you refer to by array name and number (called the index).

Example:

```java
int[] x = new int[10];
```

instead of

```java
int x0, x1, x2, x3, x4, x5, x6, x7, x8, x9;
```
Declaring Arrays

- Must declare, just like any other variable.
- Declarations look like creating objects:
  
  ```java
  Scanner s = new Scanner(System.in);
  ```

  ```java
  int[] x = new int[10];
  ```
Declaring Arrays

- Must declare, just like any other variable.
- Declarations look like creating objects:
  \[
  \text{Scanner } s = \text{ new Scanner(System.in)};
  \]
  \[
  \text{int[]} x = \text{ new int[10]};
  \]

  Type of the variable
Declaring Arrays

- Must declare, just like any other variable.
- Declarations look like creating objects:
  
  ```java
  Scanner s = new Scanner(System.in);
  ```

  ```java
  int[] x = new int[10];
  ```

  The variable name. In both cases, these are reference variables.
Declaring Arrays

- Must declare, just like any other variable.
- Declarations look like creating objects:

```java
Scanner s = new Scanner(System.in);
```

```java
int[] x = new int[10];
```

The keyword `new` to allocate memory
Declaring Arrays

- Must declare, just like any other variable.
- Declarations look like creating objects:
  ```java
  Scanner s = new Scanner(System.in);
  int[] x = new int[10];
  ```

The constructor for the object or array
Declaring Arrays

Syntax:

\[ \text{type[]} \ \text{variableName} = \text{new type[size]}; \]

Examples:

\[
\begin{align*}
\text{double[]} \ \text{dataSet} &= \text{new double[30]}; \\
\text{int[]} \ \text{quizScores} &= \text{new int[5]}; \\
\text{String[]} \ \text{args} &= \;
\end{align*}
\]
Using Arrays

- Syntax:
  
  `variableName[index-expression]`

- If var is an array (reference) variable of type `type[]`, then `var[expr]` is a variable of type `type`.

- Examples:
  
  ```java
double[] dataSet = new double[30];
dataSet[0] = dataSet[1]/2;
```
Array Length

- Syntax: \( \text{variableName}.length \) (note no parentheses)

- If \( \text{var} \) is an array (reference) variable, then \( \text{var}.length \) is an \textit{int} containing number of entries in array.

Examples:

\[
\text{double[]} \ \text{dataSet} = \text{new double}[30]; \\
\text{dataSet.length} == 30
\]

- Note that indices go \( 0 \) .. \( \text{dataSet.length}-1! \)
Teen Talk Barbie™

- In 1993, Mattel introduced Teen Talk Barbie™, a doll programmed to speak random phrases. Each doll had a random selection of 4 phrases from a list that included:
  - You can never have enough clothes.
  - Let’s go shopping!
  - Math class is tough.
  - (Talking GI Joe doll said “Vengeance is mine!” and “Dead men tell no lies.”)
Customizable Talking Doll

- Imagine a customizable doll instead.
- When first turned on, parents put in five phrases.
- After that, each time activates doll, it randomly says one of the phrases.

- How would you program this?
  - For simplicity, let’s just use strings.