Getting Started with Java
Recitation – 1/23/2009

CS 180
Department of Computer Science,
Purdue University
Project 1

- Now posted on the class webpage.
- Due Wed, Jan. 28 at 10 pm.
- Start early!
- All questions on the class newsgroup.
- Evening consulting hours from Monday to Wednesday during 7-10 p.m. in LWSN B146.
How to Solve This?

Problem statement:
- Write a program that asks for the user’s first, middle, and last names and replies with their initials.
- Example:
  - input: Andrew Lloyd Weber
  - output: ALW

How do you understand this problem?
- Input restraints or error tolerance?
- Ask once or multiple times?
- …
Overall Plan

- Identify the major tasks the program has to perform.
  - We need to know what to develop before we develop!

- Tasks:
  - Get the user’s first, middle, and last names
  - Extract the initials and create the monogram
  - Output the monogram
Development Steps

- We will develop this program in two steps:
  - Start with the program template and add code to get input
  - Add code to compute and display the monogram

- Any more step in real life?
  - Do not forget to test every part of your program
  - Debug and improve your program
Step 1 Design

- The program specification states “get the user’s name” but doesn’t say how.

- How to get input?
  - Use JOptionPane (standard class)
  - Input Style Choice #1
    - Input first, middle, and last names separately
  - Input Style Choice #2
    - Input the full name at once
  - We choose Style #2 because it is easier and quicker for the user to enter the information
Why Use Standard Classes

- Don’t reinvent the wheel. When there are existing classes that satisfy our needs, use them.
- Learning how to use standard Java classes is the first step toward mastering OOP.
- Before we can learn how to define our own classes, we need to learn how to use existing classes.
JOptionPane for Output

- Using `showMessageDialog` of the `JOptionPane` class is a simple way to bring up a window with a message.

```java
JOptionPane.showMessageDialog(null, "How are you?");
```

- How to show multiple lines of text?
  - Another line: "\n"
JOptionPane for Input

- Using `showInputDialog` of the `JOptionPane` class is another way to input a string.

```java
JOptionPane.showMessageDialog(null, "Your full name:");
```
String

- The textual values passed to the `showMessageDialog` method are instances of the `String` class.
- A sequence of characters separated by double quotes is a `String` constant.
- There are close to 50 methods defined in the `String` class. We will introduce three of them here: `substring`, `length`, and `indexOf`.
- We will also introduce a string operation called concatenation.
Usage of String Object

- Declaration
  ```java
  String name;
  ```

- Creation
  ```java
  name = new String(“Jane Java”);
  ```

- We can combine them together
  ```java
  String name = new String(“Jane Java”);
  ```

- Indexing from 0 to length-1
  - Referring to the string `name`, which character’s index is 3?
String Methods

- Assume `str` is a String object and properly initialized to “Purdue!”.
- Substring: `str.substring(i, j)`
  - What is `str.substring(1, 3)`?
- Length: `str.length()`
  - What is `str.length()`?
- Substring: `str.indexOf(substr)`
  - What is `str.indexOf("ue")`?
- Concatenation: `str1 + str2`
  - What is “Hi! “ + str?
- Refer to Java API or lecture slides for more information
Chapter 2 Sample Program: Displays the Monogram
File: Step1/Ch2Monogram.java

import javax.swing.*;

class Ch2Monogram {
    public static void main (String[ ] args) {
        String name;
        name = JOptionPane.showInputDialog(null,
                "Enter your full name (first,
                middle, last): ");
        JOptionPane.showMessageDialog(null, name);
    }
}
Step 1 Test

- In the testing phase, we run the program and verify that
  - we can enter the name
  - the name we enter is displayed correctly

- Why do we test before finishing the whole problem?
  - How to find a small bug in a large room?
  - What about finding a small bug on a small piece of paper?
Step 2 Design

- Our programming skills are limited, so we will make the following assumptions:
  - input string contains first, middle, and last names
  - first, middle, and last names are separated by single blank spaces

- Example
  - John Quincy Adams (okay)
  - John Kennedy (not okay)
  - Harrison, William Henry (not okay)
Step 2 Design

- Given the valid input, we can compute the monogram by
  - breaking the input name into first, middle, and last
  - extracting the first character from them
  - concatenating three first characters

```
“Aaron Ben Cosner”
```
```
<table>
<thead>
<tr>
<th>“Aaron”</th>
<th>“Ben Cosner”</th>
</tr>
</thead>
<tbody>
<tr>
<td>“Ben”</td>
<td>“Cosner”</td>
</tr>
<tr>
<td>“ABC”</td>
<td></td>
</tr>
</tbody>
</table>
```
Chapter 2 Sample Program: Displays the Monogram
File: Step1/Ch2Monogram.java

import javax.swing.*;

class Ch2Monogram {
    public static void main (String[] args) {
        String name, first, middle, last,
            space, monogram;

        space = " " ;
        //Input the full name
        name = JOptionPane.showInputDialog(null,
            "Enter your full name (first,
            middle, last) :");
    }
}"
Step 2 Code

//Extract first, middle, and last names
first = name.substring(0, name.indexOf(space));
name = name.substring(name.indexOf(space)+1,
    name.length());
middle = name.substring(0, name.indexOf(space));
last = name.substring(name.indexOf(space)+1,
    name.length());

//Compute the monogram
monogram = first.substring(0, 1) +
    middle.substring(0, 1) +
    last.substring(0,1);

//Output the result
JOptionPane.showMessageDialog(null,
    "Your monogram is " + monogram);
Step 2 Test

- In the testing phase, we run the program and verify that, for all valid input values, correct monograms are displayed.
- We run the program numerous times. Seeing one correct answer is not enough. We have to try out many different types of (valid) input values.
Program Review

- The work of a programmer is not done yet.
- Once the working program is developed, we perform a critical review and see if there are any missing features or possible improvements.
- One suggestion
  - Improve the initial prompt so the user knows the valid input format requires single spaces between the first, middle, and last names.
- Any other suggestion?
More Standard Classes

- **Standard output**: `System.out.println(...)`
  ```java
  System.out.println("Welcome to\nPurdue");
  ```
- **Standard input**: `System.in`
- **Scanner**
  ```java
  Scanner scanner = new Scanner(System.in);
  ```
- **Date**
  ```java
  Date today = new Date();
  System.out.println(today.toString() + "\n");
  ```
  - **SimpleDateFormat**
- **Refer to Java API or lecture slides for more information**
Coding Style

- Take a careful look at the coding standards on the class website
- Develop or keep your own good coding style
- Good for readers, good for yourself
Quiz

- Write some code to print the following stuff:
  
  Hey!
  Well done!
  
  - Hint: `System.out.print(...)`

- Declare a String object `school` and let its value be "Purdue University".