CS 112 Introduction to Programming

Variable Scoping;
Nested Loops;
Parameterized Methods

Yang (Richard) Yang
Computer Science Department
Yale University
308A Watson, Phone: 432-6400
Email: yry@cs.yale.edu
Review Basic Concepts
Item 1: Implement a Method that takes a string and a number, and print the string number of times

(switch to Eclipse view)
Item 2: Finish the Coding and Testing DrawX (we finished the pseudo code; see class schedule page for the current progress: DrawXPartial emphasis on testing)
Nested Loop Design Example: Drawing A Complex Pattern

- Use nested for loops to draw ASCII X
- A size SIZE ASCII X has 2 * SIZE rows
- Why draw ASCII art?
  - Real graphics will require some finesse (shortly)
  - ASCII art has complex patterns
  - Can focus on the algorithms
Final Pseudo Code

1. Bound
   • == , (SIZE – 2)*2 spaces, ==

2. for line = 1 to SIZE -1
   line spaces
   \  
   2 (SIZE – 2) + 2 – 2 * line spaces
   /

3. for line = 1 to SIZE – 1
   SIZE – line spaces
   /  
   (line – 1) * 2 spaces
   \  

4. Bound
Item 2: Potential Exercise
Exercise: Box Figures

- Design Boxes.java to print the following boxes:

```
******
*   *
******

********
*        *
********

******
*   *
*   *
******
```
public static void main(String[] args) {
    drawBox(5, 3);
    drawBox(10, 3);
    drawBox(5, 4);
}

public static void repeat(String s, int times) {
    for (int i = 1; i <= times; i++) {
        System.out.print(s);
    }
}

public static void drawBox(int width, int height) {
    // step 1. print width of "*"
    repeat("*", width);
    System.out.println();
    // step 2. loop a total of height - 2 times
    // for each row:
    //   print *
    //   print width -2 " "
    for (int i = 1; i <= height - 2; i++) {
        System.out.print("*");
        repeat(" ", width - 2);
        System.out.println("*");
    }
    // step 3. print width of "*"
    repeat("*", width);
    System.out.println();
}

} // end of main
Procedural Heuristics

1. The main method should read as a concise summary of the overall set of tasks performed by the program.

2. Each method should have a clear set of responsibilities.

3. No method should do too large a share of the overall task.

4. Use method with parameters to remove redundancy with generalization, but do not over generalize.

5. Data should be declared/used at the lowest level possible (localized data).
Item 3: DrawMatrix
Design Exercise

- What nested for loops produce the following output?

Inner loop, *parameterized* behavior.

\[
\begin{align*}
&\ldots 1 \\
&\ldots 2 \\
&\ldots 3 \\
&. 4 \\
&5
\end{align*}
\]

- We must build multiple complex lines of output using:
  - an *outer* "vertical" loop for each of the lines
  - *inner* "horizontal" loop(s) for the patterns within each line
Outer and inner loop

- First write the outer loop, from 1 to the number of lines.

```
for (int line = 1; line <= 5; line++) {
...
}
```

- Now look at the line contents. Each line has a pattern:
  - some dots (0 dots on the last line), then a number

```
....1
...2
..3
 .4
 5
```

- Observation: the number of dots is related to the line number.
Nested for loop solution

- Answer:
  ```java
  for (int line = 1; line <= 5; line++) {
      for (int j = 1; j <= (-1 * line + 5); j++) {
          System.out.print(".");
      }
      System.out.println(line);
  }
  ```

- Output:
  ```plaintext
  ....1
  ...2
  ..3
  .4
  5
  ```
What is the output of the following nested for loops?

```java
int N = 5;
for (int line = 1; line <= N; line++) {
    for (int j = 1; j <= (-1 * line + N); j++) {
        System.out.print(".");
    }
    for (int k = 1; k <= line; k++) {
        System.out.print(line);
    }
    System.out.println();
}
```

Answer:

```
....1
...22
..333
 .4444
 55555
```
Practice: Nested for loop exercise

- Write code to produce diagonal matrix:
  
  ....1
  ...2.
  ..3..
  .4...
  5....
Practice: Nested for loop exercise

- Write code to produce diagonal matrix:
  ....1
  ...2.
  ..3..
  .4...
  5....

- Answer:
  ```java
  int N = 5;
  for (int line = 1; line <= N; line++) {
      for (int j = 1; j <= (-1 * line + N); j++) {
          System.out.print(".");
      }
      System.out.print(line);
      for (int j = 1; j <= (line - 1); j++) {
          System.out.print(".");
      }
      System.out.println();
  }
  ```
Exercise: Change the `DrawMatrix` program to use a parameterized method for different sizes.

**DrawMatrixMethods.java**
Item 4: DrawMirror
Design Example: Drawing A Complex Pattern

- Use nested for loops to produce the following output

- Why draw ASCII art?
  - Real graphics require some finesse
  - ASCII art has complex patterns
  - Can focus on the algorithms

```java
#================#
|      <><>      |
|    <>....<>    |
|  <>........<>  |
|<>............<>|
|<>............<>|
|  <>........<>  |
|    <>....<>    |
|      <><>      |
#================#
```

Mirror.java
Pseudo-code algorithm

1. Line
   • # , 16 =, #

2. Top half
   • |   #
     • spaces (decreasing)
     • <>
     • dots (increasing)
     • <>
     • spaces (same as above)
     • |

3. Bottom half (top half upside-down)

4. Line
   • # , 16 =, #
public class Mirror {
    public static void main(String[] args) {
        line();
        topHalf();
        bottomHalf();
        line();
    }

    public static void topHalf() {
        for (int line = 1; line <= 4; line++) {
            // contents of each line
        }
    }

    public static void bottomHalf() {
        for (int line = 1; line <= 4; line++) {
            // contents of each line
        }
    }

    public static void line() {
        // ...
    }
}
## 2. Tables

- **A table for the top half:**
  - **Compute spaces and dots expressions from line number**

<table>
<thead>
<tr>
<th>line</th>
<th>spaces</th>
<th>line * -2 + 8</th>
<th>dots</th>
<th>4 * line - 4</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>6</td>
<td>6</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>2</td>
<td>4</td>
<td>4</td>
<td>4</td>
<td>4</td>
</tr>
<tr>
<td>3</td>
<td>2</td>
<td>2</td>
<td>8</td>
<td>8</td>
</tr>
<tr>
<td>4</td>
<td>0</td>
<td>0</td>
<td>12</td>
<td>12</td>
</tr>
</tbody>
</table>
3. Writing the code

- Useful questions about the top half:
  - What methods? (think structure and redundancy)
  - Number of (nested) loops per line?
// Prints the expanding pattern of <> for the top half of the figure.
public static void topHalf() {
    for (int line = 1; line <= 4; line++) {
        System.out.print("|");
        for (int space = 1; space <= (line * -2 + 8); space++) {
            System.out.print(" ");
        }
        System.out.print("}><");
        for (int dot = 1; dot <= (line * 4 - 4); dot++) {
            System.out.print(".");
        }
        System.out.print("><");
        for (int space = 1; space <= (line * -2 + 8); space++) {
            System.out.print(" ");
        }
        System.out.println("|");
    }
}
SimpleStdDraw.java Example (in Eclipse)

Useful StdDraw.java Description and Examples:
http://introcs.cs.princeton.edu/java/15inout/

Full StdDraw API Documentation:
http://zoo.cs.yale.edu/classes/cs112/cs112-2014-spring/doc_StdDraw/