CS112 PSET3 Walk-Through

Haoran (May) Wang, Qiao Xiang, Yang (Richard) Yang
Computer Science Department
Yale University
208A Watson, Phone: 432-6400
Email: cs112ta@cs.yale.edu, yry@cs.yale.edu
# Number Coolness

| 1 X 8  + 1 = 9  |
| 12 X 8  + 2 = 98 |
| 123 X 8 + 3 = 987 |
| 1234 X 8 + 4 = 9876 |
| 12345 X 8 + 5 = 98765 |
| 123456 X 8 + 6 = 987654 |
| 1234567 X 8 + 7 = 9876543 |
| 12345678 X 8 + 8 = 98765432 |
| 123456789 X 8 + 9 = 987654321 |

Output of `printNumberCoolness(9)`

- Implement the `printNumberCoolness` method
- What patterns do we observe?
  - # of lines
  - # of spaces in each line
  - First number
  - “ X 8 + ” in each line
  - Third number
  - “ = “ in each line
  - Last number
- What do we do to generate these patterns?
Mapping Loop# to Target Pattern

<table>
<thead>
<tr>
<th>Loop#</th>
<th>Output of printNumberCoolness(9)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1 x 8 + 1 = 9</td>
</tr>
<tr>
<td></td>
<td>12 x 8 + 2 = 98</td>
</tr>
<tr>
<td></td>
<td>123 x 8 + 3 = 987</td>
</tr>
<tr>
<td></td>
<td>1234 x 8 + 4 = 9876</td>
</tr>
<tr>
<td></td>
<td>12345 x 8 + 5 = 98765</td>
</tr>
<tr>
<td></td>
<td>123456 x 8 + 6 = 987654</td>
</tr>
<tr>
<td></td>
<td>1234567 x 8 + 7 = 9876543</td>
</tr>
<tr>
<td></td>
<td>12345678 x 8 + 8 = 98765432</td>
</tr>
<tr>
<td></td>
<td>123456789 x 8 + 9 = 987654321</td>
</tr>
</tbody>
</table>

- Loop# (x) → # of lines

<table>
<thead>
<tr>
<th>Loop#</th>
<th># of lines</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td>4</td>
<td>4</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
</tr>
</tbody>
</table>

- # of lines = loop#

```java
for (int i = 1; i <= N; i++) {
    //print one line in each iteration.
}
```
Mapping Loop# to Target Pattern

\[
\begin{align*}
1 \times 8 + 1 &= 9 \\
12 \times 8 + 2 &= 98 \\
123 \times 8 + 3 &= 987 \\
1234 \times 8 + 4 &= 9876 \\
12345 \times 8 + 5 &= 98765 \\
123456 \times 8 + 6 &= 987654 \\
1234567 \times 8 + 7 &= 9876543 \\
12345678 \times 8 + 8 &= 98765432 \\
123456789 \times 8 + 9 &= 987654321 \\
\end{align*}
\]

Output of `printNumberCoolness(9)`

- Loop# \( x \) → # of spaces in each line
- # of lines = 9 – loop#
- Replace 9 with N
- # of lines = N – loop#
Mapping Loop# to Target Pattern

• Loop# (x) → First number

<table>
<thead>
<tr>
<th>Loop#</th>
<th># of spaces in each line</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>2</td>
<td>12 = 1*10+2</td>
</tr>
<tr>
<td>3</td>
<td>123 = 12*10+3</td>
</tr>
<tr>
<td>4</td>
<td>1234 = 123*10 + 4</td>
</tr>
<tr>
<td>…</td>
<td>…</td>
</tr>
</tbody>
</table>

• First number = first number from last line * 10 + loop#

• Other than loop#, need to remember the first number from last loop!

• How to do that in your program?

Output of printNumberCoolness(9)

1 \times 8 + 1 = 9
12 \times 8 + 2 = 98
123 \times 8 + 3 = 987
1234 \times 8 + 4 = 9876
12345 \times 8 + 5 = 98765
123456 \times 8 + 6 = 987654
1234567 \times 8 + 7 = 9876543
12345678 \times 8 + 8 = 98765432
123456789 \times 8 + 9 = 987654321
Mapping Loop# to Target Pattern

- Loop# (x) → " X 8 + " in each line

<table>
<thead>
<tr>
<th>Loop#</th>
<th>“ X 8 + ”</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>“ X 8 + ”</td>
</tr>
<tr>
<td>2</td>
<td>“ X 8 + ”</td>
</tr>
<tr>
<td>3</td>
<td>“ X 8 + ”</td>
</tr>
<tr>
<td>4</td>
<td>“ X 8 + ”</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
</tr>
</tbody>
</table>

• “ X 8 + ” shows once in every line!

Output of printNumberCoolness(9)

1 × 8 + 1 = 9
12 × 8 + 2 = 98
123 × 8 + 3 = 987
1234 × 8 + 4 = 9876
12345 × 8 + 5 = 98765
123456 × 8 + 6 = 987654
1234567 × 8 + 7 = 9876543
12345678 × 8 + 8 = 98765432
123456789 × 8 + 9 = 987654321
## Mapping Loop# to Target Pattern

- **Loop# (x) → Third number**
- **Loop# (x) → “ = ”**
- **Loop# (x) → Last number**

You can figure out pretty quickly.

<table>
<thead>
<tr>
<th>Loop# (x)</th>
<th>Target Pattern</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1 X 8 + 1 = 9</td>
</tr>
<tr>
<td>12</td>
<td>12 X 8 + 2 = 98</td>
</tr>
<tr>
<td>123</td>
<td>123 X 8 + 3 = 987</td>
</tr>
<tr>
<td>1234</td>
<td>1234 X 8 + 4 = 9876</td>
</tr>
<tr>
<td>12345</td>
<td>12345 X 8 + 5 = 98765</td>
</tr>
<tr>
<td>123456</td>
<td>123456 X 8 + 6 = 987654</td>
</tr>
<tr>
<td>1234567</td>
<td>1234567 X 8 + 7 = 9876543</td>
</tr>
<tr>
<td>12345678</td>
<td>12345678 X 8 + 8 = 98765432</td>
</tr>
<tr>
<td>123456789</td>
<td>123456789 X 8 + 9 = 987654321</td>
</tr>
</tbody>
</table>

**Output of printNumberCoolness(9)**
The 5-Semi-Colon Challenge

• Additional hints
  • What happens when you do “+” on Strings?
  • Does the order matter in the increment part of the for loop?

• Questions about Number Coolness?
Rocket Ship

• Implement the `drawRocket` method
• What approach should we use?
• Top-Down Decomposition
• What is a good decomposition for ASCII drawing?

“ASCII drawing cursor movement constraint: left -> right, top -> bottom. So it is better to decompose along movement line.” -- CS112 Lecture 8

Output of `drawRocket(3)`
Decompose a Rocket Ship

Triangle

Connector

Upperbody

Connector

Lowerbody

Connector

Triangle
Can we do better?

Upperbody

<table>
<thead>
<tr>
<th>Upperbody</th>
<th>TwinPeak</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>TwinPeak</td>
</tr>
</tbody>
</table>

Lowerbody

<table>
<thead>
<tr>
<th>Lowerbody</th>
<th>Inverse TwinPeak</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Inverse TwinPeak</td>
</tr>
</tbody>
</table>

TwinPeak

Inverse TwinPeak

Inverse TwinPeak

TwinPeak
Final Decomposition

Triangle

Connector

TwinPeak

Inverse TwinPeak

Connector

TwinPeak

Inverse TwinPeak

Connector

Triangle
What methods do we need?

Drawing repeated characters is a reusable function.
Implementing each method

• Observe target patterns
• Map loop# to target patterns

• Questions about Rocket Ship?
CafeWall
Implement the `drawRow` method

• What is in a row?
  • A certain number of black/white pairs of boxes

• What parameters decide a row?
  • (x, y): the coordinates of lower left corner
  • pairs: number of black/white box pairs
  • size: the size of each box
Can we do a top-down decomposition?
Implement the `drawGrid` method

• What is in a grid?
  • A certain number of rows

• What parameters decide a grid?
  • $(x, y)$: the coordinates of lower left corner
  • pairs: number of black/white box pairs
  • size: the size of each box
  • Lower row offset
Implement the `drawGrid` method
Experience in using StdDraw

• Some StdDraw methods I used:
  • filledSquare
  • line
  • setPenColor

• Read the javadoc carefully. Pay special attention to the coordinates and the size.

• Questions about CafeWall?