CS 312 – Day 2 – Extra Slides

Chand T. John
How to prepare for Quiz 1 on Monday

- Get a CS account!!! (If you don’t have one by Monday, we’ll allow you to use your own computer if you have one, just this one time.)
- Do (don’t just “review”) the practice problems in this slide deck.
- To check your work, try running your code in Eclipse (or whatever editor you chose to use). If you have a CS account, you can run code in Eclipse on a GDC 3rd or 1st floor lab computer.
- Do the Practice-It! Problems from our class website (https://cs.utexas.edu/~chand/cs312) listed for the first two days of class
- Post questions and/or discuss all of the above practice problems on Piazza
Escape sequences

• System.out.println();
  Prints a blank line
• System.out.println(“”); // other way is better style
  Prints a blank line
• System.out.println(“\n”);
  Prints two blank lines
• System.out.println(“\t”);
  Prints a tab and a new line
• System.out.println(“\”);
  Prints a backslash
• System.out.println(“\””);
  Prints a double quote
Example – What is the output of this program?

```java
public class Cs312 {
    public static void main(String[] args) {
        System.out.println("But I keep cruising\nCan’t stop won’t stop moving");
        System.out.println("It’s like I go\t this music");
        System.out.println("In my mind, saying \"it\’s gonna be alright\"");
    }
}
```
Example – What is the output of this program?

```java
public class Cs312 {
    public static void main(String[] args) {
        System.out.println("But I keep cruising\nCan’t stop won’t stop moving");
        System.out.println("It’s like I go\this music");
        System.out.println("In my mind, saying "it’s gonna be alright"");
    }
}
```

**Output is:**
But I keep cruising
Can’t stop won’t stop moving
It’s like I go this music
In my mind, saying "it's gonna be alright"

Lyrics altered from “Shake It Off” by Taylor Swift
Example – what’s the output?

System.out.println("A\nB\nC\tC\tt\nD\\n\\\\\\n\\\\\\n");

**Output:**
A
B
n  C  t
D"
"
Write a program using only System.out.println statements that produces this output:

```
  .
 / \
/   \
/_____\ 
(  _- _ ) 
( )  ( )
( )  "  " ( )
( )  .  ( )
/ )  o ( \ 
/ ) \_/\ ( \ 
/ ) \___\ ( \ 
```

Note that the highlighted characters are all on the left end of the screen (i.e., no spaces before them). No, you don’t need to print highlighted characters—that’s not something you can do with println statements in Java.
public class Cs312 {
    public static void main(String[] args) {
        System.out.println("   o");
        System.out.println("  / \");
        System.out.println(" /   \");
        System.out.println("/_____\");
        System.out.println("( _-_- )");
        System.out.println("()   ()");
        System.out.println("()" "()");
        System.out.println("/) o (\");
        System.out.println("/)\_/(\");
        System.out.println("  ---");
    }
}

public class Cs312 {
    public static void main(String[] args) {
        System.out.println("Twinkle, twinkle, little star, ");
        System.out.println("How I wonder what you are! ");
        System.out.println("Up above the world so high, ");
        System.out.println("Like a diamond in the sky. ");
        System.out.println("Twinkle, twinkle, little star, ");
        System.out.println("How I wonder what you are! ");
    }
}
public class Cs312 {
    public static void main(String[] args) {
        System.out.println("Twinkle, twinkle, little star,");
        System.out.println("How I wonder what you are!");
        System.out.println("Up above the world so high,");
        System.out.println("Like a diamond in the sky.");
        System.out.println("Twinkle, twinkle, little star,");
        System.out.println("How I wonder what you are!");
    }
}

Repetitive code
public class Cs312 {
    public static void main(String[] args) {
        chorus();
        System.out.println("Up above the world so high,");
        System.out.println("Like a diamond in the sky.");
        chorus();
    }

    public static void chorus() {
        System.out.println("Twinkle, twinkle, little star," );
        System.out.println("How I wonder what you are!");
    }
}
Every positive integer is the sum of powers of 2

- $72 = 64 + 8 = 2^6 + 2^3$
- $83 = 64 + 16 + 2 + 1 = 2^6 + 2^4 + 2^1 + 2^0$
- $59 = ?$
Write a program whose output is the following body of text repeated exactly 59 times:

Who run the world?
Girls!
Who run the world?
Girls!

Use only System.out.println statements and methods you define to minimize repetition in your code.

Lyrics from “Run the World” by Beyoncé Knowles
public class Cs312 {
    public static void main(String[] args) {
        output32();
        output16();
        output8();
        output2();
        output1();
    }

    public static void output32() {
        output16();
        output16();
    }

    public static void output16() {
        output8();
        output8();
    }

    public static void output8() {
        output4();
        output4();
    }

    public static void output2() {
        output1();
        output1();
    }

    public static void output1() {
        lyrics();
        lyrics();
    }

    public static void lyrics() {
        System.out.println("Who run the world?");
        System.out.println("Girls!");
    }
}
More practice for Quiz 1
Fabric patterns

[Images of fabric patterns]


CS 312 – Chand John
Brick pattern

9 equals signs

One space

_________  ___________  ___________  ___________

====  ___________  ___________  ___________  ====

_________  ___________  ___________  ___________

====  ___________  ___________  ___________  ====

_________  ___________  ___________  ___________

====  ___________  ___________  ___________  ====

_________  ___________  ___________  ___________
public class Cs312 {
    public static void main(String[] args) {
        System.out.println("=*=*=*=*=*=*=*=*=*=*=*=*=*=*=*=*=*=*=*");
        System.out.println("****=*=*=*=*=*=*=*=*=*=*=*=*=*=*=*=***");
        System.out.println("=*=*=*=*=*=*=*=*=*=*=*=*=*=*=*=*=*=*=*");
        System.out.println("****=*=*=*=*=*=*=*=*=*=*=*=*=*=*=*=***");
        System.out.println("=*=*=*=*=*=*=*=*=*=*=*=*=*=*=*=*=*=*=*");
        System.out.println("=*=*=*=*=*=*=*=*=*=*=*=*=*=*=*=*=*=*=*");
    }
}

Repetitive! Is there a way to do this with less repetitive typing??
public class Cs312 {
    public static void main(String[] args) {
        System.out.println("=*=*=*=*=*=*=*=*=*=*=*=*=*=*=*=*=*");
        System.out.println("=*=*=*=*=*=*=*=*=*=*=*=*=*=*=*=*=*");
        System.out.println("=*=*=*=*=*=*=*=*=*=*=*=*=*=*=*=*=*");
        System.out.println("=*=*=*=*=*=*=*=*=*=*=*=*=*=*=*=*=*");
    }
}

Repetitive! Is there a way to do this with less repetitive typing??
Brick pattern – better solution

Can we remove any more repetition?

Also, in this class we don’t really want one-line methods!
public class Cs312 {
    public static void main(String[] args) {
        brickTwoLines();
        brickTwoLines();
        System.out.println("========= ========= ========= =========");
    }

    public static void brickTwoLines() {
        System.out.println("========= ========= ========= =========");
        System.out.println("==== ========= ========= ========= ====");
    }
}
Diagonal pattern
Can you write a program that prints out a diagonal pattern with exactly 100 lines of output?
What would 100 lines look like?

These 5 lines of text repeat over and over!
So 100 lines would just repeat the same 5 lines, 20 times!

Using the power-of-2 decomposition method:

\[ 20 = 16 + 4 \]
public class Cs312 {
    public static void main(String[] args) {
        output16();
        output4();
    }

    public static void output16() {
        output8();
        output8();
    }

    public static void output8() {
        output4();
        output4();
    }

    public static void output4() {
        output2();
        output2();
    }

    public static void output2() {
        output1();
        output1();
    }

    public static void output1() {
        System.out.println("\====\====\====\" );
        System.out.println("=\====\====\====" );
        System.out.println("==\====\====\===" );
        System.out.println("===\====\====\==" );
        System.out.println("====\====\====\="");
    }
}

20 = 16 + 4
Other solutions are possible

• For Quiz 1, you will be asked to use this powers-of-2 decomposition method for reducing repetition using static methods.

• However, there are many other ways to break up problems like this... if you find a pattern that is more efficient than the powers of 2 strategy, feel free to use it! But be aware that if the way you break up your program is noticeably less efficient than the powers of 2 strategy, you could lose some points on Quiz 1.
public class Cs312 {
  public static void main(String[] args) {
    output16();
    output4();
  }

  public static void output16() {
    output4();
    output4();
    output4();
    output4();
  }

  public static void output4() {
    output1();
    output1();
    output1();
    output1();
  }

  public static void output1() {
    System.out.println("\====\====\====\=");
    System.out.println("=\====\====\====");
    System.out.println("==\====\====\===");
    System.out.println("===\====\====\==");
    System.out.println("====\====\====\=");
  }
}
Diamond pattern

---X-------X-------X---
--XXX------XXX------XXX--
-XXXXXX-XXXXXX-XXXXXX-
XXXXXXXXXXXXXXXXXXXXXXXXXXXX
-XXXXXX-XXXXXX-XXXXXX-
--XXX------XXX------XXX--
---X-------X-------X---
--XXX------XXX------XXX--
-XXXXXX-XXXXXX-XXXXXX-
XXXXXXXXXXXXXXXXXXXXXXXXXXXX
-XXXXXX-XXXXXX-XXXXXX-
--XXX------XXX------XXX--
---X-------X-------X---
---X-------X-------X---
Can you write a program that prints out a diamond pattern with exactly 100 lines of output?
What would 100 lines look like?

---X-----X-----X-----
---XXX----XXX----XXX---
-XXXXX-XXXXX-XXXXX-
XXXXXXXXXXXXXXXXXXX
-XXXXX-XXXXX-XXXXX-
---XXX---XXX---XXX--
---X-----X-----X---
---XXX---XXX---XXX--

This pattern repeats over and over!

6 lines * 16 = 96, then need 4 more lines, can't include a full 6-line pattern at the end.
public class Cs312 {
    public static void main(String[] args) {
        wholePattern16();
        patternTop4();
    }

    public static void wholePattern16() {
        wholePattern8();
        wholePattern8();
    }

    public static void wholePattern8() {
        wholePattern4();
        wholePattern4();
    }

    public static void wholePattern4() {
        wholePattern2();
        wholePattern2();
    }

    public static void wholePattern2() {
        wholePattern();
        wholePattern();
    }

    public static void wholePattern() {
        patternTop4();
        patternBottom2();
    }

    public static void patternTop4() {
        System.out.println("---X-----X-----X---");
        System.out.println("--XXX---XXX---XXX--");
        System.out.println("-XXXXX-XXXXX-XXXXX-");
        System.out.println("XXXXXXXXXXXXXXXXXXX");
    }

    public static void patternBottom2() {
        System.out.println("-XXXXX-XXXXX-XXXXX-"cls
        System.out.println("--XXX---XXX---XXX--");
    }
}
public class Cs312 {
    public static void main(String[] args) {
        wholePattern16();
        patternTop4();
    }

    public static void wholePattern16() {
        wholePattern4();
        wholePattern4();
        wholePattern4();
        wholePattern4();
    }

    public static void wholePattern4() {
        wholePattern();
        wholePattern();
        wholePattern();
        wholePattern();
    }

    public static void wholePattern() {
        patternTop4();
        patternBottom2();
    }

    public static void patternTop4() {
        System.out.println("---X-----X-----X---");
        System.out.println("--XXX---XXX---XXX--");
        System.out.println("-XXXXX-XXXXX-XXXXX-");
        System.out.println("XXXXXXXXXXXXXXXXXXX");
    }

    public static void patternBottom2() {
        System.out.println("-XXXXX-XXXXX-XXXXX-");
        System.out.println("--XXX---XXX---XXX--");
    }
}