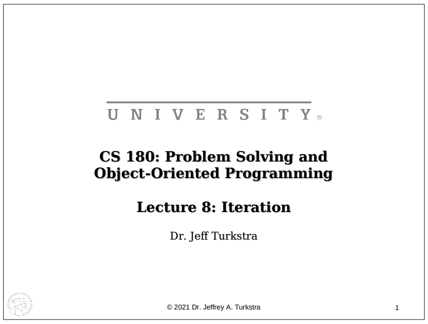
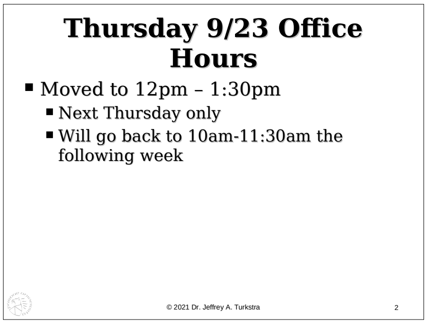


UNIVERSITY

CS 180: Problem Solving and Object-Oriented Programming

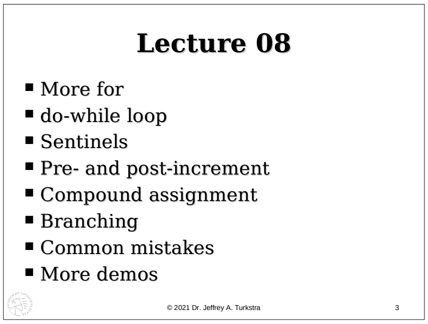
Lecture 8: Iteration

Dr. Jeff Turkstra



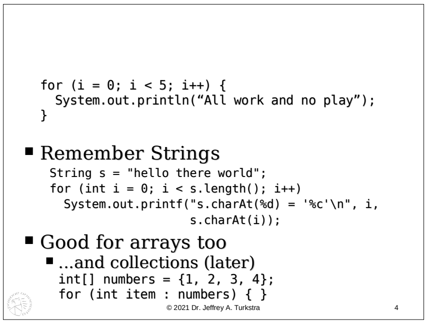
Thursday 9/23 Office Hours

- Moved to 12pm – 1:30pm
  - Next Thursday only
  - Will go back to 10am-11:30am the following week



Lecture 08

- More for
- do-while loop
- Sentinels
- Pre- and post-increment
- Compound assignment
- Branching
- Common mistakes
- More demos



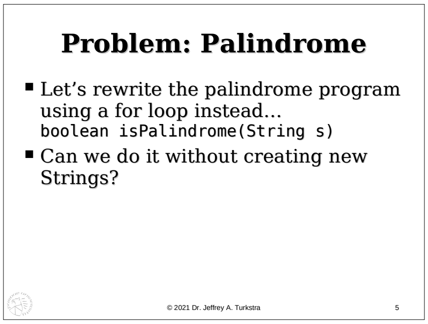
```
for (i = 0; i < 5; i++) {
  System.out.println(“All work and no play”);
}
```

- Remember Strings

```
String s = "hello there world";
for (int i = 0; i < s.length(); i++)
  System.out.printf("s.charAt(%d) = '%c'\n", i,
                    s.charAt(i));
```

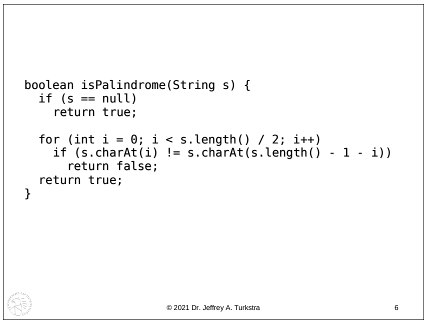
- Good for arrays too
  - …and collections (later)

```
int[] numbers = {1, 2, 3, 4};
for (int item : numbers) { }
```

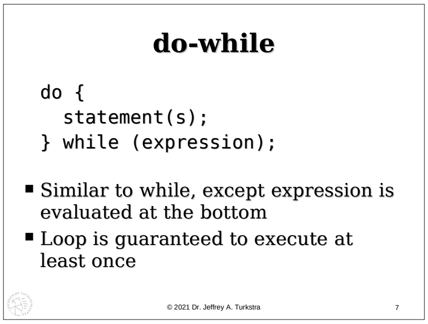


Problem: Palindrome

- Let’s rewrite the palindrome program using a for loop instead… `boolean isPalindrome(String s)`
- Can we do it without creating new Strings?



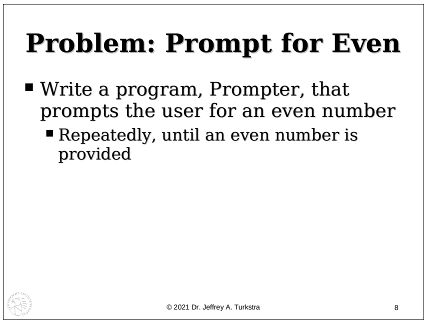
```
boolean isPalindrome(String s) {
  if (s == null)
    return true;

  for (int i = 0; i < s.length() / 2; i++)
    if (s.charAt(i) != s.charAt(s.length() - 1 - i))
      return false;
  return true;
}
```



do-while

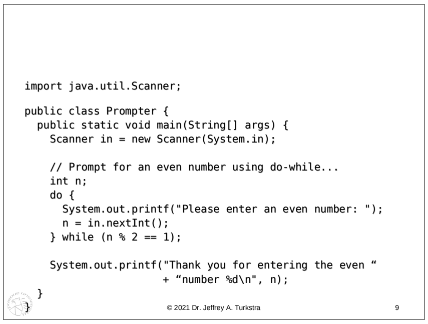
```
do {
  statement(s);
} while (expression);
```

- Similar to while, except expression is evaluated at the bottom
- Loop is guaranteed to execute at least once

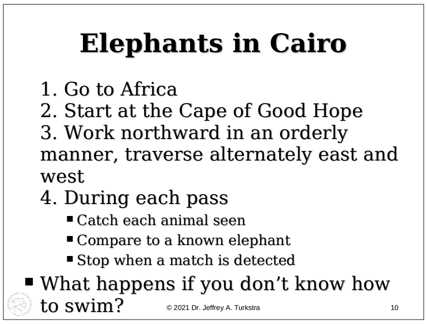


Problem: Prompt for Even

- Write a program, Prompter, that prompts the user for an even number
  - Repeatedly, until an even number is provided



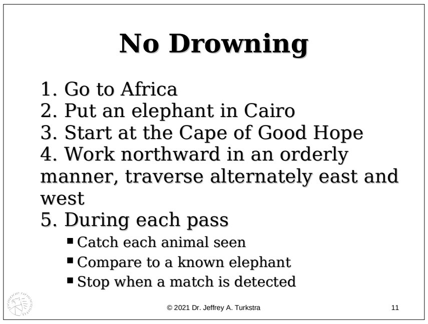
```
import java.util.Scanner;

public class Prompter {
  public static void main(String[] args) {
    Scanner in = new Scanner(System.in);

    // Prompt for an even number using do-while...
    int n;
    do {
      System.out.printf("Please enter an even number: ");
      n = in.nextInt();
    } while (n % 2 == 1);

    System.out.printf("Thank you for entering the even “
                  + “number %d\n", n);
  }
}
```



Elephants in Cairo

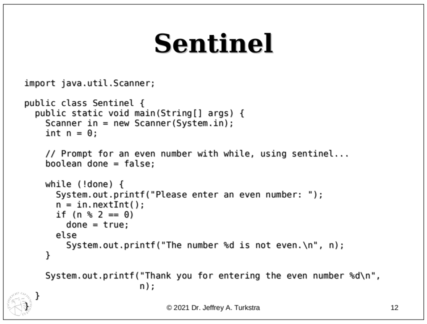
1. Go to Africa
2. Start at the Cape of Good Hope
3. Work northward in an orderly manner, traverse alternately east and west
4. During each pass
   - Catch each animal seen
   - Compare to a known elephant
   - Stop when a match is detected

- What happens if you don’t know how to swim?



No Drowning

1. Go to Africa
2. Put an elephant in Cairo
3. Start at the Cape of Good Hope
4. Work northward in an orderly manner, traverse alternately east and west
5. During each pass
   - Catch each animal seen
   - Compare to a known elephant
   - Stop when a match is detected



Sentinel

```
import java.util.Scanner;

public class Sentinel {
  public static void main(String[] args) {
    Scanner in = new Scanner(System.in);
    int n = 0;

    // Prompt for an even number with while, using sentinel...
    boolean done = false;

    while (!done) {
      System.out.printf("Please enter an even number: ");
      n = in.nextInt();
      if (n % 2 == 0)
        done = true;
      else
        System.out.printf("The number %d is not even.\n", n);
    }

    System.out.printf("Thank you for entering the even number %d\n",
                  n);
  }
}
```

A note on incrementing

- x++, post-increment operator
  - Increments x by one, but expression value is the original x
- ++x, pre-increment operator
  - Increments x by one, and expression value is the new x

Multiple statements

```
int i;
int sum;

for (i = 0, sum = 0; i <= 10; sum += i, i++);

System.out.println(“Sum is: “ + sum);
```

Multiplication table

```
int row, col;

for (col = 1; col < 13; col++) {
  System.out.print(“ “ + col);
}
System.out.print(“\n”);

for (row = 1; row < 13; row++) {
  System.out.print(row);

  for (col = 1; col < 13; col++) {
    System.out.print(“ “ + row * col);
  }
  System.out.print(“\n”);
}
```

Compound assignment

- Common assignment statements:
```
x = x + y;
a = a – b;
s = s + “\n”;    // s is a String
```
- Java provides a shortcut:
```
x += y;
a -= b;
s += “\n”;
```
- Works for all (most) binary operators

Purdue Trivia

- Purdue is home to Indiana’s first and only nuclear reactor
  - Built in 1962
  - Built by Lockheed Corporation
  - Three stories beneath the Duncan Annex of EE
  - Criticality on August 30, 1962
  - Dedication September 27

Branching statements

- break
- continue
- return

break

- Remember switch
- Also works for loops
- Means “terminate this switch, for, while, or do-while statement”

- Can use labeled statements to break out of nested loops
  - Not the same as goto!
  - Execution resumes after the designated loop statement

continue

- Skip the current iteration of a for, while, or do-while loop
- Also has labeled and unlabeled forms
- Always executes the boolean control expression next

return

- Exit current method and return to where it was invoked
- Optional value

Common mistakes

- Infinite loop
- Fencepost errors
- Skipped loop
- Misplaced semicolon

Problem: Divisor

- Print an NxN table that indicates whether the given row and column values divide each other

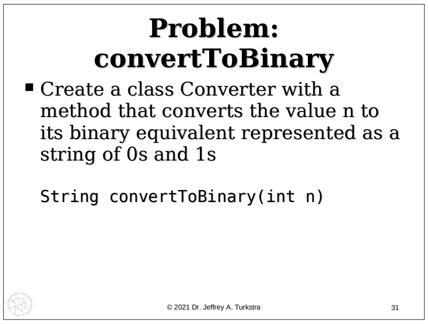
```
   1  2  3  4  5
1  *  *  *  *  *
2  *  *     *
3  *     *
4  *  *     *
5  *           *
```

```
public class DivisorPattern {
  public static void main(String[] args) {
    int N = Integer.parseInt(args[0]);

    System.out.printf("  ");
    for (int i = 1; i <= N; i++)
      System.out.printf("%3d", i);
    System.out.printf("\n");

    for (int i = 1; i <= N; i++) {
      System.out.printf("%3d", i);
      for (int j = 1; j <= N; j++) {
        if (i % j == 0 || j % i == 0)
          System.out.printf("  *");
        else
          System.out.printf("   ");
      }
      System.out.printf("\n");
    }
  }
}
```

Problem: Collatz

- Compute the Hailstone sequence for a given number n and track how many iterations it takes to reach 1
  - If n is odd, multiply by 3 and add 1
  - If n is even, divide by 2
  - Repeat with the new value of n

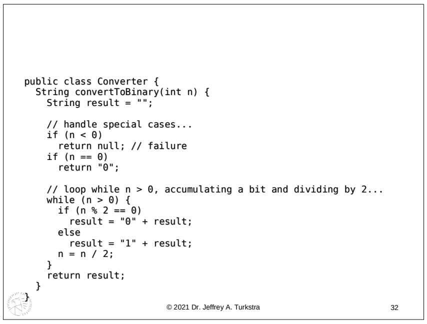
```
public class Collatz {
  int count(int n) {
    int c = 1;
    while (n > 1) {
      c++;
      if (n % 2 == 0)
        n /= 2;
      else
        n = 3 * n + 1;
    }
    return c;
  }

  public static void main(String[] args) {
    Collatz c = new Collatz();
    System.out.println(c.count(22));
    System.out.println(c.count(1));
    System.out.println(c.count(1000));
  }
}
```



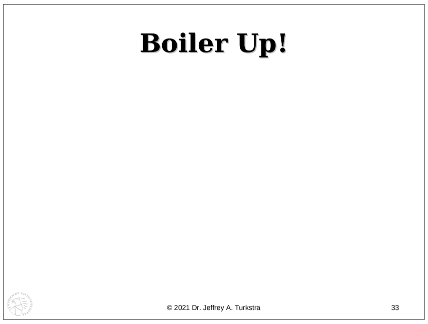
Problem: convertToBinary

- Create a class Converter with a method that converts the value n to its binary equivalent represented as a string of 0s and 1s

  ```
  String convertToBinary(int n)
  ```



```
public class Converter {
  String convertToBinary(int n) {
    String result = "";

    // handle special cases...
    if (n < 0)
      return null; // failure
    if (n == 0)
      return "0";

    // loop while n > 0, accumulating a bit and dividing by 2...
    while (n > 0) {
      if (n % 2 == 0)
        result = "0" + result;
      else
        result = "1" + result;
      n = n / 2;
    }
    return result;
  }
}
```



Boiler Up!