“A fine balance must be maintained between computation time and the ensuing complexity of the coding.”

Real Admiral Grace Hopper, Ph.D. and Howard Aiken, Ph.D.

“A Manual of Operation for the Automatic Sequence Controlled Calculator”
Nested if/else question

Formula for body mass index (BMI):

\[ BMI = \frac{weight}{height^2} \times 703 \]

Write a program that produces output like the following:

This program reads data for two people and computes their body mass index (BMI) and weight status.

Enter next person's information:
height (in inches)? 73.5
weight (in pounds)? 230
BMI = 29.93
overweight

Enter next person's information:
height (in inches)? 71
weight (in pounds)? 220.5
BMI = 30.75
obese

Difference = 0.82

<table>
<thead>
<tr>
<th>BMI</th>
<th>Weight class</th>
</tr>
</thead>
<tbody>
<tr>
<td>below 18.5</td>
<td>underweight</td>
</tr>
<tr>
<td>18.5 - 24.9</td>
<td>normal</td>
</tr>
<tr>
<td>25.0 - 29.9</td>
<td>overweight</td>
</tr>
<tr>
<td>30.0 and up</td>
<td>obese</td>
</tr>
</tbody>
</table>
Easy but bad design (1 of 2)

import java.util.Scanner;

public class Cs312Class2 {
    public static void main(String[] args) {
        Scanner key = new Scanner(System.in);
        System.out.println("This program reads data for two people and computes");
        System.out.println("their body mass index (BMI) and weight status.");
        System.out.println();
        System.out.println("Enter next person's information:");
        System.out.println("Enter height (in inches)? ");
        double height1 = key.nextDouble();
        System.out.println("Enter weight (in pounds)? ");
        double weight1 = key.nextDouble();
        double bmi1 = weight1 / height1 / height1 * 703.0;
        System.out.printf("BMI = %.2f
", bmi1);

        if (bmi1 < 18.5) {
            System.out.println("underweight");
        } else if (bmi1 < 25.0) {
            System.out.println("normal");
        } else if (bmi1 < 30.0) {
            System.out.println("overweight");
        } else {
            System.out.println("obese");
        }
        System.out.println();
System.out.println("Enter next person's information:");
System.out.print("height (in inches)? ");
double height2 = key.nextDouble();
System.out.print("weight (in pounds)? ");
double weight2 = key.nextDouble();
double bmi2 = weight2 / height2 / height2 * 703.0;
System.out.printf("BMI = \%.2f\n", bmi2);

if (bmi2 < 18.5) {
    System.out.println("underweight");
} else if (bmi2 < 25.0) {
    System.out.println("normal");
} else if (bmi2 < 30.0) {
    System.out.println("overweight");
} else {
    System.out.println("obese");
}
System.out.println();

double difference = Math.abs(bmi2 - bmi1);
System.out.printf("Difference = \%.2f\n", difference);
Procedural heuristics

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

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

3. Minimize coupling and dependencies between methods.

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

5. Variables should be declared/used at the lowest level possible.
import java.util.Scanner;

public class Cs312Class2 {
    public static void main(String[] args) {
        Scanner key = new Scanner(System.in);

        introduction();

        double bmi1 = getBmi(key);
        printBmiResults(bmi1);

        double bmi2 = getBmi(key);
        printBmiResults(bmi2);

        printBmiDifference(bmi1, bmi2);
    }

    public static void introduction() {
        System.out.println("This program reads data for two people and computes");
        System.out.println("their body mass index (BMI) and weight status.");
        System.out.println();
    }

    public static double getBmi(Scanner key) {
        System.out.println("Enter next person's information:");
        double height = getDouble(key, "height (in inches)?");
        double weight = getDouble(key, "weight (in pounds)?");
        double bmi = weight / height / height * 703.0;
        return bmi;
    }
}
public static void printBmiResults(double bmi) {
    System.out.printf("BMI = %.2f\n", bmi);
    System.out.println(bmiCategory(bmi));
    System.out.println();
}

public static String bmiCategory(double bmi) {
    if (bmi < 18.5) {
        return "underweight";
    } else if (bmi < 25.0) {
        return "normal";
    } else if (bmi < 30.0) {
        return "overweight";
    } else {
        return "obese";
    }
    return "obese";
}

public static double getDouble(Scanner key, String promptWithoutSpace) {
    System.out.print(promptWithoutSpace + " ");
    return key.nextDouble();
}

public static void printBmiDifference(double bmi1, double bmi2) {
    double difference = Math.abs(bmi2 - bmi1);
    System.out.printf("Difference = %.2f\n", difference);
}
Indexes

- Characters of a string are numbered with 0-based *indexes*:

  String name = "K. Scott";

<table>
<thead>
<tr>
<th>index</th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
</tr>
</thead>
<tbody>
<tr>
<td>character</td>
<td>K</td>
<td>.</td>
<td>S</td>
<td>c</td>
<td>o</td>
<td>t</td>
<td>t</td>
<td></td>
</tr>
</tbody>
</table>

- First character's index : 0 (zero based indexing)
- Last character's index : 1 less than the string's length
- The individual characters are values of type `char` (another primitive data type)
### String methods

<table>
<thead>
<tr>
<th>Method name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>indexOf(str)</code></td>
<td>index where the start of the given string appears in this string (-1 if not found)</td>
</tr>
<tr>
<td><code>length()</code></td>
<td>number of characters in this string</td>
</tr>
<tr>
<td><code>replace(str1, str2)</code></td>
<td>replaces occurrences of <code>str1</code> with <code>str2</code></td>
</tr>
<tr>
<td><code>substring(index1, index2)</code> or <code>substring(index1)</code></td>
<td>the characters in this string from <code>index1</code> (inclusive) to <code>index2</code> (exclusive); if <code>index2</code> is omitted, grabs till end of string</td>
</tr>
<tr>
<td><code>toLowerCase()</code></td>
<td>a new string with all lowercase letters</td>
</tr>
<tr>
<td><code>toUpperCase()</code></td>
<td>a new string with all uppercase letters</td>
</tr>
</tbody>
</table>

- These methods are called using the dot notation:

```java
String student = "Olivia Scott";
System.out.println(student.length());  // 12
```
String method examples

// index 012345678901
String s1 = "Olivia Scott";
String s2 = "Isabelle Scott";
System.out.println(s2.length()); // 14
System.out.println(s1.indexOf("e")); // -1
System.out.println(s2.indexOf("e")); // 4
System.out.println(s1.substring(7, 10)); // "Sco"
String s3 = s2.substring(4, 10); // "elle s"
System.out.println(s3.toLowerCase()); // "elle s"

Given the following string:

// index 0123456789012345678901
String book = "Building Java Programs";

- How would you extract the word "Building"?
  (Write code that can extract the first word from any string.)
Clicker Question 1

What is output by the following code?

```java
String s1 = "Football";
String s2 = s1.substring(4, 8);
s2.substring(1);
System.out.print(s2);
```

A. Football  
B. ball  
C. all  
D. No output due to syntax error.  
E. No output due to runtime error.
Modifying strings

- Methods like `substring` and `toLowerCase` build and return a new string, rather than modifying the current string.

```java
String s = "ut Longhorns";
s.toUpperCase();
System.out.println(s);    // ut Longhorns
```

- To modify a variable's value, you must reassign it:

```java
String s = "ut Longhorns";
s = s.toUpperCase();
System.out.println(s);    // UT LONGHORNS
```
Strings as user input

- **Scanner's `next` method** reads a word of input as a `String`.

  ```java
  Scanner console = new Scanner(System.in);
  System.out.print("What is your first name? ");
  String name = console.next();
  System.out.println(name + " has " + name.length() + " letters and starts with " + name.substring(0, 1));
  ```

  Output:
  What is your first name? **Chamillionaire**
  Chamillionaire has 14 letters and starts with C

- **The `nextLine` method** reads a line of input as a `String`.

  ```java
  System.out.print("What is your address? ");
  String address = console.nextLine();
  ```
Clicker Question 2

What is output by the following code?

```java
String s1 = "taxicab";
String s2 = "acables";
String s3 = s1.substring(4);
String s4 = s2.substring(1, 4);
if(s3.length() == s4.length())
    System.out.println("1");
else
    System.out.println("2");
if(s3 == s4)
    System.out.println("1");
else
    System.out.println("2");
```

A. 11  
B. 12  
C. 21  
D. 22  
E. No output due to syntax error
Comparing strings

- Relational operators such as < and == fail on objects.

```java
Scanner console = new Scanner(System.in);
System.out.print("What is your name? ");
String name = console.next();
if (name == "Barney") {
    System.out.println("I love you, you love me,\n    We're a happy family!");
}
```

- This code will compile, but it will not print the song.
- == compares objects by references (seen later), so it often gives false even when two Strings have the same letters.
The **equals method**

- Objects are compared using a method named `equals`.
  
  ```java
  Scanner console = new Scanner(System.in);
  System.out.print("What is your name? ");
  String name = console.next();
  if (name.equals("Barney")) {
      System.out.println("Fred's Friend.");
      System.out.println("Purple Dinasuar.");
      System.out.println("In trouble.");
  }
  ```

- The **equals method** returns a value of type boolean, the type used in logical tests.
String test methods

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>equals(str)</td>
<td>whether two strings contain the same characters</td>
</tr>
<tr>
<td>equalsIgnoreCase(str)</td>
<td>whether two strings contain the same characters, ignoring upper vs. lower case</td>
</tr>
<tr>
<td>startsWith(str)</td>
<td>whether one contains other's characters at start</td>
</tr>
<tr>
<td>endsWith(str)</td>
<td>whether one contains other's characters at end</td>
</tr>
<tr>
<td>contains(str)</td>
<td>whether the given string is found within this one</td>
</tr>
</tbody>
</table>

```java
String name = console.next();
if (name.startsWith("Prof")) {
    System.out.println("When are your office hours?");  
} else if (name.endsWith("OBE")) {
    System.out.println("Yes Sir!");
}
```
Strings questions

- Write a method to determine if a String is a possible representation of a DNA strand
  - contains only A, C, T, and G

- Write a method to create a *Watson-Crick complement* given a String that represents a strand of DNA
  - replace A with T, C with G, and vice versa

- Given a String that represents a strand of DNA, return the first substring that exists between "ATG" and either "TAG" or "TGA"
  - no overlap allowed
Write a method that returns the number of times a given character occurs in a String.

Write a method that returns the number of times the punctuation marks . ? ! , : " ; ' occur in a String.