“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 earnings:

\[
\text{dollarsEarned} = \text{dollarsPerHour} \times \text{hoursWorked}
\]

Write a program that produces output like the following:

This program reads data for two people and reacts to their earnings.

Enter next person's info:
pay rate (in dollars per hour)? 1
hours worked? 1
Work more!

Enter next person's info:
pay rate (in dollars per hour)? 10
hours worked? 10
Good job!

Earnings difference = $99.00
import java.util.Scanner;

public class Cs312 {
    public static void main(String[] args) {
        Scanner key = new Scanner(System.in);
        System.out.println("This program reads data for two people");
        System.out.println("and reacts to their earnings.");
        System.out.println();

        System.out.println("Enter next person's info: ");
        System.out.print("pay rate (in dollars per hour)? ");
        double dollarsEarnedPerHour1 = key.nextDouble();
        System.out.print("hours worked? ");
        double hoursWorked1 = key.nextDouble();
        double dollarsEarned1 = dollarsEarnedPerHour1 * hoursWorked1;

        if (dollarsEarned1 < 10.0) {
            System.out.println("Work more!");
        } else if (dollarsEarned1 < 100.0) {
            System.out.println("Good, but work even more.");
        } else if (dollarsEarned1 < 1000.0) {
            System.out.println("Good job!");
        } else if (dollarsEarned1 < 10000.0) {
            System.out.println("Great job!");
        } else {
            System.out.println("$$$$$$$$$$!");
        }
        System.out.println();
    }
}
System.out.println("Enter next person's info: ");
System.out.print("pay rate (in dollars per hour)? ");
double dollarsEarnedPerHour2 = key.nextDouble();
System.out.print("hours worked? ");
double hoursWorked2 = key.nextDouble();
double dollarsEarned2 = dollarsEarnedPerHour2 * hoursWorked2;

if (dollarsEarned2 < 10.0) {
    System.out.println("Work more!");
} else if (dollarsEarned2 < 100.0) {
    System.out.println("Good, but work even more.");
} else if (dollarsEarned2 < 1000.0) {
    System.out.println("Good job!");
} else if (dollarsEarned2 < 10000.0) {
    System.out.println("Great job!");
} else {
    System.out.println("$$$$$$$$$$!");
}
System.out.println();

double difference = Math.abs(dollarsEarned2 - dollarsEarned1);
System.out.printf("Earnings difference = $%.2f\n", difference);

key.close();
}
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.
```java
import java.util.Scanner;

public class Cs312 {
    public static void main(String[] args) {
        Scanner key = new Scanner(System.in);
        introduction();

        double dollarsEarned1 = getDollarsEarned(key);
        printResults(dollarsEarned1);

        double dollarsEarned2 = getDollarsEarned(key);
        printResults(dollarsEarned2);

        printDifference(dollarsEarned1, dollarsEarned2);
        key.close();
    }

    public static void introduction() {
        System.out.println("This program reads data for two people");
        System.out.println("and reacts to their earnings.");
        System.out.println();
    }

    public static double getDollarsEarned(Scanner key) {
        System.out.println("Enter next person's info: ");
        double dollarsEarnedPerHour1 = getNextValue(key, "pay rate (in dollars per hour)?");
        double hoursWorked1 = getNextValue(key, "hours worked?");
        return dollarsEarnedPerHour1 * hoursWorked1;
    }
```
Better solution, cont'd.

```java
public static double getNextValue(Scanner key, String promptNoSpace) {
    System.out.print(promptNoSpace + " ");
    return key.nextDouble();
}

public static void printResults(double dollarsEarned) {
    System.out.println(reaction(dollarsEarned));
    System.out.println();
}

public static String reaction(double dollarsEarned) {
    if (dollarsEarned < 10.0) {
        return "Work more!";
    } else if (dollarsEarned < 100.0) {
        return "Good, but work even more.";
    } else if (dollarsEarned < 1000.0) {
        return "Good job!";
    } else if (dollarsEarned < 10000.0) {
        return "Great job!";
    } else {
        return "$$$$$$$$$$!";
    }
}

public static void printDifference(double dollarsEarned1, double dollarsEarned2) {
    double difference = Math.abs(dollarsEarned2 - dollarsEarned1);
    System.out.printf("Earnings difference = $%.2f\n", difference);
}
```
Better ➔ easier to change

import java.io.PrintStream;
import java.io.UnsupportedEncodingException;
import java.util.Scanner;

public class Cs312 {
    public static void main(String[] args) throws UnsupportedEncodingException {
        Scanner key = new Scanner(System.in);
        introduction();

        double dollarsEarned1 = getDollarsEarned(key);
        printResults(dollarsEarned1);

        double dollarsEarned2 = getDollarsEarned(key);
        printResults(dollarsEarned2);

        printDifference(dollarsEarned1, dollarsEarned2);
        key.close();
    }

    public static void introduction() {
        System.out.println("This program reads data for two people");
        System.out.println("and reacts to their earnings.");
        System.out.println();
    }

    This is just for fun; uses some stuff we won’t cover in this class!
Better ➔ easier to change

```java
public static double getDollarsEarned(Scanner key) {
    System.out.println("Enter next person's info: ");
    double dollarsEarnedPerHour1 = getNextValue(key, "pay rate (in dollars per hour)?");
    double hoursWorked1 = getNextValue(key, "hours worked?");
    return dollarsEarnedPerHour1 * hoursWorked1;
}

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

public static void printResults(double dollarsEarned) throws UnsupportedEncodingException {
    PrintStream out = new PrintStream(System.out, true, "UTF-8");
    out.println(reaction(dollarsEarned));
    System.out.println();
}

This is just for fun; uses some stuff we won’t cover in this class!
```
public static String reaction(double dollarsEarned) {
    if (dollarsEarned < 10.0) {
        return "\u2639";
    } else if (dollarsEarned < 100.0) {
        return "\u263A";
    } else if (dollarsEarned < 1000.0) {
        return "\u266C";
    } else if (dollarsEarned < 10000.0) {
        return "\u2600";
    } else {
        return "\u265B";
    }
}

public static void printDifference(double dollarsEarned1, double dollarsEarned2) {
    double difference = Math.abs(dollarsEarned2 - dollarsEarned1);
    System.out.printf("Earnings 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>indexOf(str)</td>
<td>index where the start of the given string appears in this string (-1 if not found)</td>
</tr>
<tr>
<td>length()</td>
<td>number of characters in this string</td>
</tr>
<tr>
<td>replace(str1, str2)</td>
<td>replaces occurrences of str1 with str2</td>
</tr>
<tr>
<td>substring(index1, index2) or substring(index1)</td>
<td>the characters in this string from index1 (inclusive) to index2 (exclusive); if index2 is omitted, grabs till end of string</td>
</tr>
<tr>
<td>toLowerCase()</td>
<td>a new string with all lowercase letters</td>
</tr>
<tr>
<td>toUpperCase()</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.)
What is output by the following code?
String s1 = "Football";
String s2 = s1.substring(4, 8);
s2.substring(1);
System.out.println(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();
```
What is output by the following code?

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


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
String Questions

- 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.