CS102
Introduction to data structures, algorithms, and object-oriented programming

DAY 4
Using the acm library

Later in the semester, when we are programming with graphics and animated games, you will use the acm library again.

For now, it will be better to use the built-in Java input and output classes.
println, in the System.out class, allows you to output a String.

```
System.out.println("Welcome to Vassar College!");
```

This method puts in a newline after printing the String. You can use the line `System.out.println();` with no String argument to put a blank line in your output.

```
System.out.print("This is a line with no newline after it.");
```

is a method that does not put in a newline. The problem with this is that it brings up the cursor immediately after the output.
System.out.println and print

Inside the String argument to println, any variable concatenated to a String using the + operator is automatically turned into a String.

    System.out.println("x is "+x+" and y is "+y);

At times, concatenating the empty String first causes all subsequent values to be converted to their String equivalent:

    char beta = 'B';
    String letters = "; // set letters to empty string
    letters += "+beta+" ;

printf, in the System.out class, allows you to specify the number of places after the decimal point when printing a double. The printf method’s first argument is a String that contains a format string. The format string is embedded in the String argument as %1.2f as shown below and in Section 2.4.1 of our textbook.

```
System.out.printf("Your BMI is %1.2f.\n", bMI);
```

The \n embeds a newline after the String is printed. The , separates the String with embedded format characters from the variable to be embedded in the String.

printf can be used to format any type of data.
Scanner, Sect.2.4.6

Scanner is a class in the java.util package that provides methods that take input from the keyboard. To use a Scanner to read from the keyboard (standard input), you need to instantiate an object of type Scanner.

```java
Scanner in = new Scanner(System.in);
System.out.println("Please enter a number.");
double rnum = in.nextDouble();
System.out.printf("Number in dollars is $%1.2f.\n", rnum);
```

The keyword new is used to create a new object of type Scanner.

The input is done by calling the nextDouble instance method of the new Scanner object.

System.in refers to the keyboard.
Class in the javax.swing package that provides another way of taking input from the keyboard. To use a JOptionPane to read from the keyboard (standard input), you need to call a static method in the JOptionPane class:

```java
String in = JOptionPane.showInputDialog
    ("Please enter a line of text");
```

This causes a pop-up dialog box to appear with a blank for input.

For output, use another static method from the JOptionPane class:

```java
JOptionPane.showMessageDialog
    (null,"The answer is 42");
```

null is the default value for any object type. In graphics programs, this is often a JFrame.
Iterative Control Structures (Ch. 3)

- **for** loop*
- **while** loop*
- **do while** loop*

* indicates the keywords are directly followed by ()s
Blocks

Statements inside a loop are grouped by enclosing them in \{\}s, a block.

Blocks can contain any number of statements, including 0.

As a matter of good programming style, you should write one statement per line, use indentation to indicate statements that are contained inside a block, and include vertical space between parts of the program.
A for loop repeats a statement or multiple statements a fixed number of times. Use of a for loop is shown below:

```java
int sum = 0;
int n = Integer.parseInt(JOptionPane.showInputDialog("Enter number: "));

for(int i = 1; i <= n; i++) {
    sum += i;
}
JOptionPane.showMessageDialog(null, "Sum of numbers from 1 to " + n + " is "+sum);
```
for loops

The initialization is done inside the parenthesis at the beginning of the loop.

A loop control variable is declared, initialized, tested, and modified in the for loop statement.

```java
int sum = 0;
int n = 100;
for(int i = 1; i < n, i++)
{
    sum += i;
}
```
String str; // Line of text entered by the user.
int count;   // Number of different letters found in str.
char letter; // A letter of the alphabet.

str = JOptionPane.showInputDialog("Please type in a line of text.");
str = str.toUpperCase(); // call on non-static method in object str
count = 0;   // initialize count
JOptionPane.showMessageDialog(null,"Your input contains the following letters:");

for ( letter = 'A'; letter <= 'Z'; letter++ )
{
    int i; // Position of a character in str.
    for ( i = 0; i < str.length(); i++ ) {
        if ( letter == str.charAt(i) ) {
            System.out.print(letter);
            System.out.print(' ');
            count++;
            break;
        }
    }
}
A while loop will repeat a statement only so long as a specified condition (boolean expression) remains true. A while loop has the form:

```
while (boolean-expression) {
  // 1. b-e
  inner block statements  // 2. b-e true
}
```

// 3. b-e false

1. boolean-expression (b-e) is evaluated
2. if b-e is true, evaluate inner block statements
3. if b-e is false, start evaluating statements after block
while flow of control diagrams

while

The initialization is done outside the body of the loop.

A loop control variable is modified inside the body of the while. Eventually, the condition becomes false and the while loop ends.
while flow of control diagrams

```c
int sum = 0;
int n = 100;
int i = 1;

while(i < n) {
    sum += i;
    i++;
}
```
Loop and a half using break

A loop that stops when a positive integer is entered.

```java
int n;
while (true) {
    n = Integer.parseInt( JOptionPane.showInputDialog( "Enter a positive integer: ") );
    if (n > 0) {
        break;
    }
    System.out.println("Try again.");
} // end of while loop
JOptionPane.showMessageDialog(null, "The number is "+ n);
```
Strings

Strings are sequences of characters. Methods we will use in lab today on Strings include length, toUpperCase, charAt, indexOf:

"abcdefg".length() returns 7

"tomorrow".toUpperCase() returns "TOMORROW"

String petString = "cats and dogs";

petString.charAt(6) returns 'n'

petString.indexOf('o') returns 10

petString.indexOf('X') returns -1
if Decision Statement

1. **if ...else**: “either-or” type statement, each with its own block of code.

2. **if** alone with a block of code, only runs block if the expression is true.

3. **if, else if, else if, ..., else**. Multi-way decision statement, each part with its own block of code.

4. **?:** Short form of if...else.

if and else are like cond in Racket. Only one clause in the group is executed and the rest are ignored. The else at the end is like that in the cond, sort of a default condition.
If..Else Flow of Control

Yes

Do statement 1

Is condition true?

No

Do statement 2

if

else
String line = JOptionPane.showInputDialog("Please enter a line of text: ");
int count = line.length();
println("Your input contains \"+count+
((count>1) ? " letters.\n": " letter. \n"));

This code snippet first reads a line of text from the user and then prints the result. The ?: operator is embedded in the String that is printed.
Step-by-step description of how to solve a problem.

Each line of human language must be broken down into a language solvable by a computer.

Developing a program from a human language form involves what is called *stepwise refinement*. That is, re-write each line into a form called *pseudocode* and then write it in a computer language.