Additional resources

- **Books**
    Hebert Schildt, Tata Mc Graw Hill
  - *Object-Oriented Programming with C++ and Java*
    Debasis Samanta, Prentice Hall of India

- **Website**
  - [https://cse.iitkgp.ac.in/~dsamanta/java/index.htm](https://cse.iitkgp.ac.in/~dsamanta/java/index.htm)
Concept of Java Programming
History of Java

- James Gosling, Mike Sheridan, and Patrick Naughton initiated the Java language project in June 1991. The small team of Sun engineers called Green Team.
- Firstly, it was called "Greentalk" by James Gosling, and file extension was .gt.
- Java was originally designed for small, embedded systems in electronic appliances like set-top boxes, but it was too advanced technology for the digital cable television industry at the time.
- After that, it was called Oak and was developed as a part of the Green project. Java team members initiated this project to develop a language for digital devices.
- Later, Java technology was incorporated by Netscape as it was suited for networking.
Why Java is named Java?

- Java was called Oak as it is a symbol of strength and chosen as a national tree of many countries like U.S.A., France, Germany, Romania, etc.

- The team wanted something that reflected the essence of the technology: revolutionary, dynamic, lively, cool, unique, and easy to spell and fun to say.

- In 1995, Oak was renamed as Java
  - Java is an island of Indonesia where first coffee was produced (called java coffee).

- In 1995, Time magazine called Java one of the Ten Best Products of 1995.

- JDK (Java Development Kit) 1.0 released in January 23, 1996.
The Java programming is claimed as

- Simple
- Robust
- Portable
- Object-Oriented
- Secure
- High Performance
- Multithreaded
- Interpreted
- Platform Independent
- Dynamic
- Architecture Neutral
- Neutral
Java has consistently been more popular than any other programming language.
How is Java Unique?
Von Neumann architecture of computing
Programming languages

- Machine Level
  - 11010111
  - 10100101

- Assembly Level
  - ADD X Y
  - MOV Z A
  - ...

- High Level
  - #include<...>
  - main(){
    - printf("...")
  }

- Assembler
- Interpreter
- Compiler
A third generation (programming) language (3GL) is a grouping of programming languages that introduced significant enhancements to second generation languages, primarily intended to make the programming language more programmer-friendly.

English words are used to denote variables, programming structures and commands, and Structured Programming is supported by most 3GLs.

Commonly known 3GLs are FORTRAN, BASIC, Pascal, JAVA and the C-family (C, C+, C++, C#, Objective-C) of languages. Also known as a high-level programming language.
High-level Programming Principles
Function-oriented programming
## FOP versus OOP

<table>
<thead>
<tr>
<th></th>
<th>Function Oriented Programming (FOP)</th>
<th>Object Oriented Programming (OOP)</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Program organization</strong></td>
<td>Program is divided into small parts called <strong>functions</strong>.</td>
<td>Program is divided into parts called <strong>objects</strong>.</td>
</tr>
<tr>
<td><strong>Importance</strong></td>
<td>Importance is not given to data but to <strong>functions</strong>.</td>
<td>Importance is given to the data rather than procedures</td>
</tr>
<tr>
<td><strong>Approach</strong></td>
<td>FOP follows <strong>top down approach</strong>.</td>
<td>OOP follows <strong>bottom up approach</strong>.</td>
</tr>
<tr>
<td><strong>Access Specifiers</strong></td>
<td>Does not have any access specifier.</td>
<td>Has three access specifiers, namely Public, Private, Protected</td>
</tr>
<tr>
<td><strong>Data Moving</strong></td>
<td>Data can move freely from function to function in the system.</td>
<td>Objects can move and communicate with each other</td>
</tr>
<tr>
<td><strong>Maintainability</strong></td>
<td>To add new data and function is not so easy.</td>
<td>Provides an easy way to add new data and function</td>
</tr>
<tr>
<td><strong>Data Access</strong></td>
<td>Function uses global data for sharing that can be accessed freely from function to function in the system.</td>
<td>Object uses local data and can be accessed in a control manner</td>
</tr>
<tr>
<td><strong>Data Hiding</strong></td>
<td>No data hiding is possible, hence security is not possible.</td>
<td>Provides data hiding, hence secured programming is possible</td>
</tr>
<tr>
<td><strong>Overloading</strong></td>
<td>Polymorphism is not possible.</td>
<td>Polymorphism is possible.</td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td>C, Visual Basic, FORTRAN, Pascal.</td>
<td>C++, JAVA, VB.NET, C#.NET.</td>
</tr>
</tbody>
</table>
Java Programming Paradigm
Java is based on the concept of object-oriented programming. As the name suggests, at the center of it all is an object. Objects contain both data and the functionality that operates on that data. This is controlled by the following four paradigms:

- **Encapsulation**
  - **Inheritance**
    - **Information hiding**
    - **Polymorphism**
Encapsulation in Java is a process of wrapping code and data together into a single unit, for example, a capsule which is mixed of several medicines.
Encapsulation: Example

Book

- Title
- Author
- Accession No.
- Cost
- Borrower DOI

Book Methods
- Issue()
- Fine()
- Return()
- Open()
- Close()

Borrower

- Name
- Roll No.
- Address
- Marks
- Books[]

Borrower Methods
- Search Books()
- Request()
- Renew()
- Enroll()
- Exit()
Encapsulation: Example
Inheritance in Java is a mechanism in which one object acquires all the properties and behaviors of a parent object. It is an important part of OOPs (Object-Oriented Programming system).
Inheritance: Example

Title
Author
Accession No.
Cost
Borrower DOI

Open()
Close()
Issue()
Fine()
Return()

Publisher
Rack No.
Permission
Copy Type

DOP
Version
Department

Add()
Remove()

Display()
Close()
Open()
Copy()
Information hiding

Public
  Title
  Author

Protected
  Account No.

Private
  Cost

Public
  Issues()
  Returns()

Protected
  Resave()

Private
  Open()
  Close()
In object-oriented programming, **polymorphism** refers to a programming language's ability to process objects depending on their class.
Polymorphism: Example

print()

x, y;
s1, s2;
Img, Doc, Doc1, Doc2

Add(x, y)
Add(s1, s2)
Add(Img, Doc)
Add(Doc1, Doc2)

Add(x, y) : 12 +34
Add(s1 + s2) : Debasis + Samanta
Add(Img, Doc) : Image + Document
Add(Doc1, Doc2) : Document1 + Document2

Add two numbers
Merge two strings
Paste an Image to a document
Merge two documents
Java Programming Features
Features of Java programming

Java Core
- Interfaces
- I/O Handler
- Java bean
- Packages
- Exception handling

Java Applet
- Multithreading
- Java multimedia
- Windows toolkit
- Java swing

Java Internet
- Java script
- JSP
- JDBC
- Distributed programming
- Networking
Questions to think…

• Can a software be developed in Java so that it runs in any OS? Any machine?

• How a browser (e.g., Mozilla, Google Chrome, Safari, etc.) works in your mobile/ Computer?
Thank You
OBJECT ORIENTED PROGRAMMING WITH JAVA

Java Programming Steps

Debasis Samanta
Department of Computer Science & Engineering
Indian Institute of Technology Kharagpur
Your First Java Program
A program in C to display message

```c
#include <stdio.h>

int main()
{
    printf("Hello, World!");
    return 0;
}
```

A program in Java to display message

```java
import java.lang.*;

class HelloWorldApp
{
    public static void main(String args[])
    {
        System.out.println("Hello, World!");
    }
}
```

**Note:** Both the languages are case sensitive
# C versus Java

<table>
<thead>
<tr>
<th>Aspects</th>
<th>C</th>
<th>Java</th>
</tr>
</thead>
<tbody>
<tr>
<td>Paradigms</td>
<td>Procedural</td>
<td>Object-oriented</td>
</tr>
<tr>
<td>Platform Dependency</td>
<td>Dependent</td>
<td>Independent</td>
</tr>
<tr>
<td>Datatypes : union, structure</td>
<td>Supported</td>
<td>Not supported</td>
</tr>
<tr>
<td>Pre-processor directives</td>
<td>Supported (#include, #define)</td>
<td>Not supported</td>
</tr>
<tr>
<td>Header files</td>
<td>Supported</td>
<td>Use packages (import)</td>
</tr>
<tr>
<td>Storage class</td>
<td>Supported</td>
<td>Not supported</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Aspects</th>
<th>C</th>
<th>Java</th>
</tr>
</thead>
<tbody>
<tr>
<td>Inheritance</td>
<td>No inheritance</td>
<td>Supported (Simple inheritance)</td>
</tr>
<tr>
<td>Pointers</td>
<td>Supported</td>
<td>No Pointers</td>
</tr>
<tr>
<td>Code translation</td>
<td>Compiled</td>
<td>Interpreted</td>
</tr>
<tr>
<td>Multi-threading and Interfaces</td>
<td>Not supported</td>
<td>Supported</td>
</tr>
<tr>
<td>Exception Handling</td>
<td>No exception handling</td>
<td>Supported</td>
</tr>
<tr>
<td>Database Connectivity</td>
<td>Not supported</td>
<td>Supported</td>
</tr>
</tbody>
</table>
Java program editing

- Any text editor can be used to write Java programs. For example,
  - In Windows
    - Notepad, Edit, Wordpad, MS-Word, etc.
  - In Unix
    - vi, emacs, gedit etc.

- Save the program
  - Save the program in a file with the name
    HelloWorldApp.java
Java program editing

prompt

Microsoft(R) Windows 98
(C)Copyright Microsoft Corp 1981-1998.
C:\WINDOWS>
// Hello world Application

class HelloWorldApp {
    public static void main ( String args[] ) {
        System.out.println("Hello Java!");
    }
}
Java program editing
Java program editing

C:\WINDOWS>cd C:\java
C:\java>dir

Volume in drive C is DB02
Volume Serial Number is F3C4-E800
Directory of C:\java

.
..  <DIR>  07-22-99 11:23p  
HELLO~1  JAV  272  07-22-99 11:23p  HelloWorldApp.java
1 file(s)  3,829 bytes
2 dir(s)  218,734,592 bytes free

C:\java>
The Java compiler (javac) converts a Java program into Java byte code

- Open a DOS shell (in Windows) or Terminal (in Unix)
- Move to the directory where your Java program has been saved
- Enter the following command to compile:

  javac HelloWorldApp.java
Java program compilation
Java program compilation

```
C:\java>dir
Volume in drive C is DB02
Volume Serial Number is F3C4-E800
Directory of C:\java
.
..          <DIR>        07-22-99 11:23p ..
HELLOW~1 JAV   272 07-23-99 12:39a HelloWorldApp.java
HELLOW~1 CLA   478 07-23-99 12:40a HelloWorldApp.class
2 file(s)     750 bytes
2 dir(s)      218,734,592 bytes free
C:\java>
```
To execute the Java program, type the command `java` (from the command prompt).

For example, the current program `HelloWorldApp.class` can be executed as

```
java HelloWorldApp
```
Java program execution

```java
C:\\java> java HelloWorldApp
Hello World!
C:\\java>
```
Java program execution

Java Program:
```java
class HelloWorldApp {
    public static void main(String[] args) {
        System.out.println("Hello World!");
    }
}
```

HelloWorldApp.java

Diagram:
- Compiler
  - Interpreter (Win32)
  - Interpreter (Solaris)
  - Interpreter (MacOS)
  - Hello World!
C/C++ versus Java execution

**C**
- **Source code (.c) file**
- **Pre-Processor**
- **Compiler**
  - Assembly code
- **Assembler**
- **Linker**
  - Object file (.o)
- **Libraries**
- **Executable file**

**THE COMPILATION PROCESS**

**At compile time**
- **.java**
  - Java source
- **Compiler**
- **.class Bytecode**

**At run time**
- **Native code**
- **JIT Compiler**
- **Back to **.class Bytecode**
C++ versus Java
Areas of applications

– **C++** is best suitable for developing large software.
  • Library management system, Employee management system, Passenger reservation system, etc.

– **Java** is best suitable for developing communication/Internet application software.
  • Network protocols, Internet programs, web page, web browser, etc.
### C++ versus Java: Programming features

<table>
<thead>
<tr>
<th>Features</th>
<th>in C++</th>
<th>in Java</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data abstraction and encapsulation</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Polymorphism</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Binding</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Static</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Dynamic</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Inheritance</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Single Inheritance</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Multiple Inheritance</td>
<td>✓</td>
<td>×</td>
</tr>
<tr>
<td>Operator overloading</td>
<td>✓</td>
<td>×</td>
</tr>
<tr>
<td>Template classes</td>
<td>✓</td>
<td>×</td>
</tr>
<tr>
<td>Global variables</td>
<td>✓</td>
<td>×</td>
</tr>
<tr>
<td>Header files</td>
<td>✓</td>
<td>×</td>
</tr>
<tr>
<td>Pointers</td>
<td>✓</td>
<td>×</td>
</tr>
<tr>
<td>Interface and packages</td>
<td>×</td>
<td>✓</td>
</tr>
<tr>
<td>API (Application Programming Interface)</td>
<td>×</td>
<td>✓</td>
</tr>
</tbody>
</table>
C++ versus Java: Programming environments

C++ provides platform dependent programming

Java provides platform independent programming
Questions to think…

• How a Java program can include two or more classes and then compile them?

• How a browser can run a Java program?
Thank You
Java Programming Tools
Tools available for Java programming

- **Java Software Developer’s Kit (SDK)**: [Java™ 2 SDK](https://java.sun.com/j2se/1.4.2/docs/index.html)
  - SDK from *JavaSoft*, a division of Sun Microsystems Inc.
  - Contains the basic tools and libraries necessary for creating, testing, documenting and executing Java programs.

- **Java™ 2 SDK, Standard Edition**
  - [https://java.sun.com/j2se/1.4.2/docs/index.html](https://java.sun.com/j2se/1.4.2/docs/index.html)
  - Official site for Java™ 2 SDK, Standard Edition
There are seven main programs in SDK

- `javac` – the Java Compiler
- `java` – the Java Interpreter
- `javadoc` – generates documentation in HTML
- `appletviewer` – the Java Interpreter to execute Java applets
- `jdb` – the Java Debugger to find and fix bugs in Java programs
- `javap` – the Java Disassembler to display the accessible functions and data in a compiled class; it also displays the meaning of byte codes
- `javah` – to create interface between Java and C routines
Tools available for Java programming

• Additional few sources
  - **Javatpoint website** : Another official site for Java™ 2 SDK, Standard Edition, help, tutorial, etc.
    
    [https://www.javatpoint.com/java-tutorial](https://www.javatpoint.com/java-tutorial)
  
  - **Free Java Download** : Download Java for your desktop computer now
    
There are many resources for learning Java

- **The Java™Tutorials**
  - The Java tutorials are practical guides for programmers who want to use the Java programming language to create applications.
    
    [https://java.sun.com/docs/books/tutorial/index.html](https://java.sun.com/docs/books/tutorial/index.html)

- **Sun Developer Network**
  - Sun Microsystem's official website listing down all the API documentation, latest Java Technologies, books and other resources.
    
    [https://java.sun.com/reference/docs/](https://java.sun.com/reference/docs/)
API (Application Programming Interface) in Java SDK

- The API enables Java programmers to develop varieties of applets and applications
- It contains **nine** packages
  - `java.applet` – for applet programming
  - `java.awt` – the Abstract Windowing Toolkit for designing GUI like Button, Checkbox, Choice, Menu, Panel, etc.
  - `java.io` – file input/output handling
  - `java.lang` – provides useful classes like to handle Object, Thread, Exception, String, System, Math, Float, Integer, etc.
Packages in Java

- **java.lang** – provides useful classes like to handle *Object, Thread, Exception, String, System, Math, Float, Integer* etc.

- **java.net** – classes for network programming; supports TCP/IP networking protocols

- **java.util** – it contains miscellaneous classes like *Vector, Stack, List, Date, Dictionary, Hash* etc.

- **javax.swing** – for designing graphical user interface (GUI)

- **java.sql** – for database connectivity (JDBC)
Other third part tools for Java programming

Java IDE (Integrated Development Environment)

– Number of IDEs are available to support the productivity of software development
  • *Sun’s Java Workshop* from Sun’s JavaSoft (recently powered with Visual Java)
  • *Mojo* from Penumbra Software (best visual environment for creating Java applets)
  • *Jumba* from Aimtech and IBM (graphical applet builder)
  • *Semantic Café* from Semantics (a de-facto standard for Java development on Windows systems)
Other third part tools for Java programming

Web browser

- Java environment requires Java-enabled web browser to support Java applets

- Few (free) popular Java-enabled web browsers:
  
  - *HotJava* from JavaSoft web site ([http://java.sun.com](http://java.sun.com))
  
  - *Netscape Navigator* from Netscape home page ([http://home.netscape.com](http://home.netscape.com))
  
  - *Internet Explorer* from Microsoft’s web page ([http://www.microsoft.com](http://www.microsoft.com))

- This is one of the most commonly used IDEs for Java and some major languages.

NotePad++ - https://notepad-plus-plus.org/download/v7.5.8.html

- This is a very advanced and handy NotePad, it has several built-in tools and functions for making programming easy.
Java Language Subset
A rich subset of the Java language

### Built-In Types
- `int`
- `long`
- `char`
- `double`
- `String`
- `boolean`

### System
- `System.out.println()`
- `System.out.print()`
- `System.out.printf()`

### Flow Control
- `if`
- `else`
- `for`
- `while`

### Parsing
- `Integer.parseInt()`
- `Double.parseDouble()`

### Boolean
- `true`
- `false`
- `||`
- `&&`
- `!`

### Arrays
- `a[i]`
- `new`
- `a.length`

### Punctuation
- `{  }
- `(  )`
- `,,;`

### String
- `+`, `...`
- `length()`, `compareTo()`
- `charAt()`, `matches()`

### Assignment
- `=`

### Objects
- `class`
- `static`
- `public`
- `private`
- `toString()`
- `equals()`
- `new`
- `main()`

### Primitive Numeric Types
- `+`, `->`, `*`
- `/`, `%`, `++`
- `--`, `>=`, `<`
- `<=`, `>=`, `==`
- `!=`

### Math Library
- `Math.sin()`
- `Math.cos()`
- `Math.log()`
- `Math.exp()`
- `Math.sqrt()`
- `Math.pow()`
- `Math.min()`
- `Math.max()`
- `Math.abs()`
- `Math.PI`
In Java, every variable has a type declared in the source code. There are two kinds of types: reference types and primitive types. Reference types are references to objects. Primitive types directly contain values.

<table>
<thead>
<tr>
<th>Type</th>
<th>Size</th>
</tr>
</thead>
<tbody>
<tr>
<td>boolean</td>
<td>1 bit</td>
</tr>
<tr>
<td>byte</td>
<td>8 bits</td>
</tr>
<tr>
<td>char</td>
<td>16 bits</td>
</tr>
<tr>
<td>short</td>
<td>16 bits</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Type</th>
<th>Size</th>
</tr>
</thead>
<tbody>
<tr>
<td>int</td>
<td>32 bits</td>
</tr>
<tr>
<td>long</td>
<td>64 bits</td>
</tr>
<tr>
<td>float</td>
<td>32 bits</td>
</tr>
<tr>
<td>double</td>
<td>64 bits</td>
</tr>
</tbody>
</table>
The Java character set

The Java language alphabet
- Uppercase letters ‘A’ to ‘Z’
- Lowercase letters ‘a’ to ‘z’
- Digits ‘0’ to ‘9’
- Java special characters:
**Identifiers in Java**

- **Identifiers**
  - Names given to various program elements (variables, constants, class, methods, etc.)
  - May consist of letters, digits and the underscore (‘_’) character, with no space between.
  - Blank and comma are not allowed.
  - First character must be an alphabet or underscore.
  - An identifier can be arbitrary long.
  - Identifier should not be a reserved word.

- **Java programming language is case sensitive.**
  - area, AREA and Area are all different!
Datatype declaration rule

Declaration and assignment statements

```java
int a, b = 0;
a = 123;
b = 45;
int c = a + b;

System.out.print("The sum is" + c);
```
Array in Java
An array is a finite, ordered and collection of homogeneous data elements.

Following are the three tasks to manipulate an array in Java

- Declaration of an array.
- Allocate memory for it.
- Loading the values into array.
Creating an array

Declaration of array

<type> <arrayName>[ ];
Example:
    int x[ ];

<type>[ ]<arrayName>;
Example:
    int [ ] x;

Allocate memory for an array

<arrayName> = new <type> [<size>];
Example:
    x = new int [100 ];
Creating an array

Define and allocate memory together

```
<type> <arrayName> [ ] = new <type> [<size>];
```

Example:

```
int x [ ] = new int [100];
```
Initialization of Array

<arrayName>[<subscript>] = <value>;

Example:

```
x[5] = 100;
```

```
for (int i = 0; i < 100; i++)
    x[i] = <value>;
```
Initialization of array: An alternative way

\[ \text{<type> <arrayName> [ ] = \{ <list of values> \};} \]

Example:
\[ \text{int x [ ] = \{12, 3, 9, 15\};} \]

Here, declaration, allocation of memory and array initialization all are at one go!
- **Insertion**
  - Insertion at any location
  - Insertion at front
  - Insertion at end
  - Insertion is sorted order

- **Deletion**
  - Deletion a particular element
  - Deletion an element at a particular location
  - Deletion the element at front
  - Deletion the element at end

- **Searching and Traversal**
  - Finding the smallest and largest element
  - Printing all element or some specific element

- **Sorting**
  - In ascending order, descending order, lexicographical order etc.
• Declaration of an array

   Examples

   ```java
   int numbers[ ];
   float averageScores[ ];
   int [ ] rollNo;
   float [ ] marks;
   ```

• Memory allocation for an array

   Examples

   ```java
   numbers = new int [5];
   averageScores = new float [20];
   rollNo = new int [49];
   marks = new float [54];
   ```

• Initialization of an array

   Examples

   ```java
   int numbers[] = {5, 4, 2, 1, 3};
   float marks[] = {2.5, 3.4, 4.5};
   ```

What is the size of the array `marks`?

```java
n = marks.length;
```
Creating a 2D array

Declare and Allocate

Example:

```csharp
int myArray [ ] [ ];
myArray = new int [3] [4];
```

OR

```csharp
int myArray [ ] [ ] = new int [3] [4];
```
Initializing a 2D array: An example

```
   1  2  3
  4  5  6
```

```
int myArray [2] [3] = {1, 2, 3, 4, 5, 6};
```

OR

```
int myArray [ ] [ ] = { {1, 2, 3}, {4, 5, 6} };
Creating a variable-sized 2D array

```csharp
<type><2DarrayName>[][] = new <type> [<rowSize>][]; for (int i = 0; i < <rowSize>; i++)
    <2DarrayName>[i] = new <type> [<colSize_i>];
```

Another way: Example

```csharp
int x [][] = new int [3][ ];
x[0] = new int [2];
x[1] = new int [4];
x[2] = new int [3];
```
### 3D arrays: An example

```java
class a3DArray {
    public static void main(String args[]) {
        int my3DArray[][][] = new int[3][4][5];
        int i, j, k;
        for(i=0; i<3; i++)
            for(j=0; j<4; j++)
                for(k=0; k<5; k++)
                    my3DArray[i][j][k] = i * j * k;
        for(i=0; i<3; i++) {
            for(j=0; j<4; j++) {
                for(k=0; k<5; k++)
                    System.out.print(my3DArray[i][j][k] + " ");
                System.out.println();
            }
            System.out.println();
        }
    }
}
```
Example program using an array

class TestArray{
    public static void main(String args[]){

        int a[] = new int[5];  //Declaration and instantiation
        a = {10, 20, 30, 40, 50};  //Initialization
        //Traversing array
        for(int i=0;i<a.length;i++){  //length is the property of array
            System.out.println(a[i]);
        }
        // Average calculation
        float sum = 0;  avg;
        for(i=0;i<a.length;i++)  //Calculating the sum of the numbers
            sum += a[i];
        avg = sum/a.length;
        System.out.println("Average = " + avg);
    }
}
Questions to think...

- How to write recursive programs in Java?
- Which program? Application or applet?
Thank You