ÖZYEĞİN UNIVERSITY

CS105
Introduction to Object-Oriented Programming

Prof. Dr. Nizamettin AYDIN

naydin@itu.edu.tr

nizamettin.aydin@ozyegin.edu.tr

Introduction

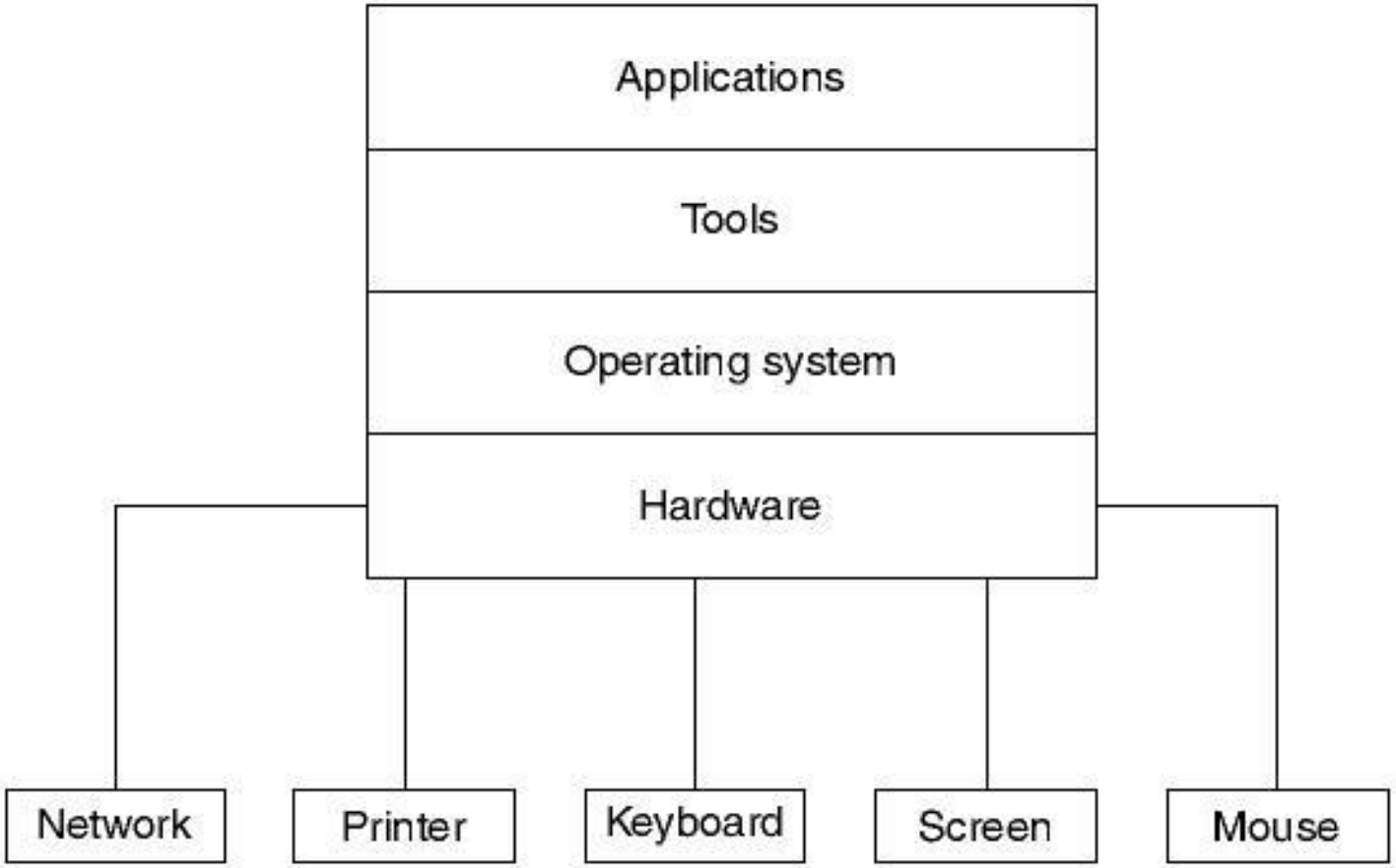
Outline

- Introduction to the Computing
- What is a Program?
- Programming Tools Overview
- Program Translation Process
- Evolution of programming languages
- Programming Language Categories
- Object-Oriented Languages
- Object-Oriented programming
- Structure of a Java Program
- Names and identifiers
- Keywords
- Syntax
- Comments
- Hello World Application

Introduction to the Computing

- In theory, computer can compute anything that's possible to compute
  - given enough memory and time
- In practice, solving problems involves computing under constraints.
  - time
    - weather forecast, next frame of animation, ...
  - cost
    - cell phone, automotive engine controller, ...
  - power
    - cell phone, handheld video game, ...

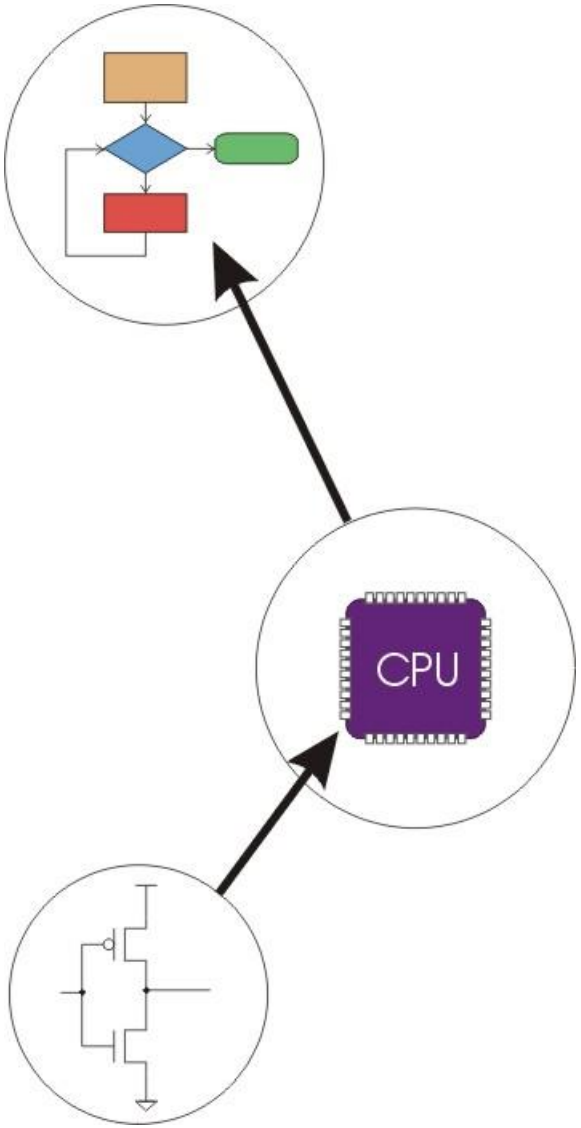
Layers of Technology



Layers of Technology

- Operating system...
  - Interacts directly with the hardware
    - Responsible for ensuring efficient use of hardware resources
- Tools...
  - Softwares that take adavantage of what the operating system has to offer.
    - Programming languages, databases, editors, interface builders...
- Applications...
  - Most useful category of software
    - Web browsers, email clients, web servers, word processors, etc...

Transformations Between Layers



Problems

Algorithms

Language

Instruction Set Architecture

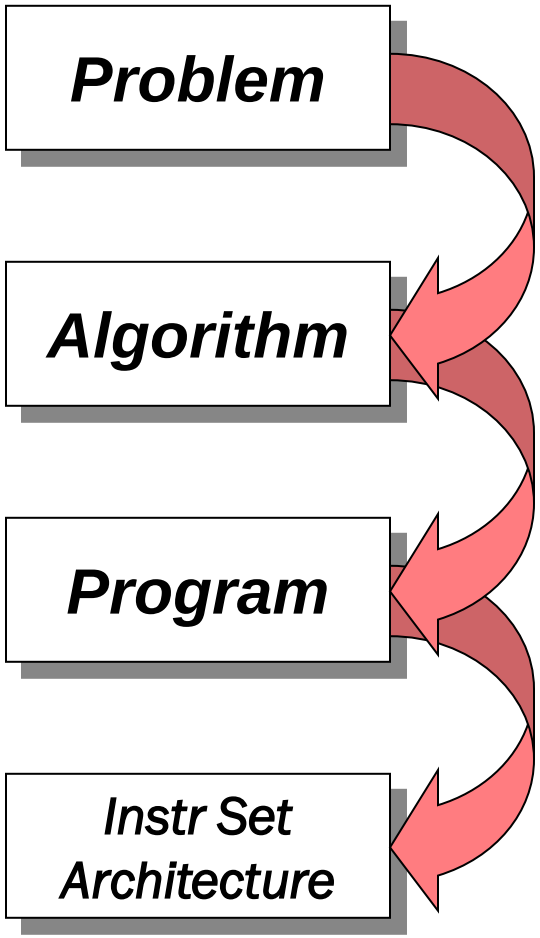
Microarchitecture

Circuits

Devices

How do we solve a problem using a computer?

- A systematic sequence of transformations between layers of abstraction.

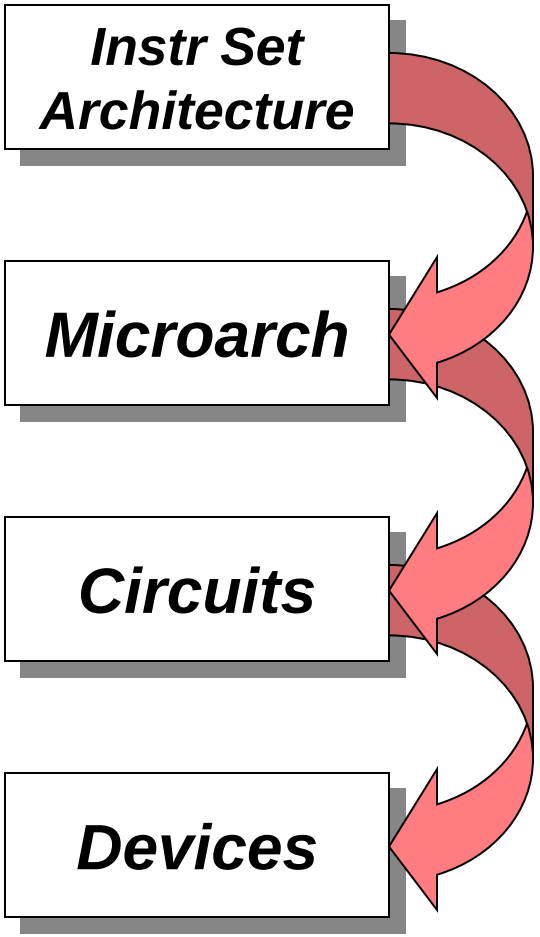


Software Design:
choose algorithms and data structures

Programming:
use language to express design

Compiling/Interpreting:
convert language to machine instructions

Deeper and Deeper…

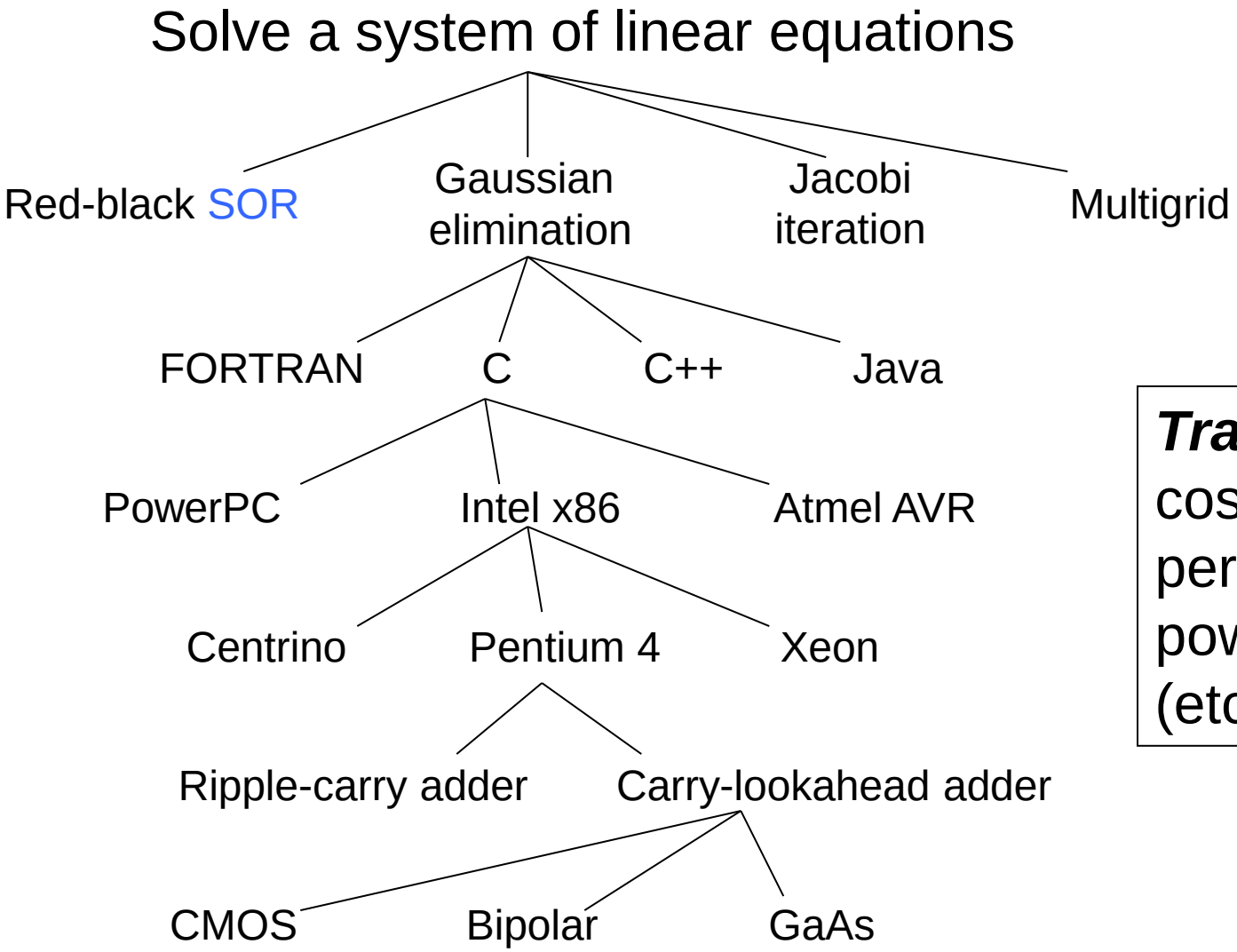


Processor Design:
choose structures to implement ISA

Logic/Circuit Design:
gates and low-level circuits to implement components

Process Engineering & Fabrication:
develop and manufacture lowest-level components

Many Choices at Each Level

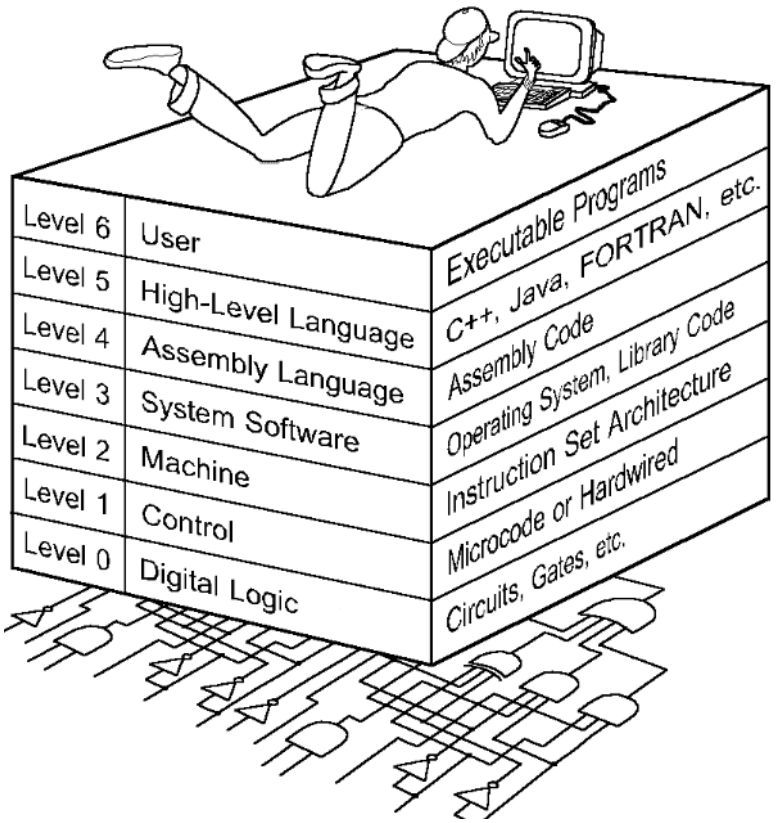


Tradeoffs:
cost
performance
power
(etc.)

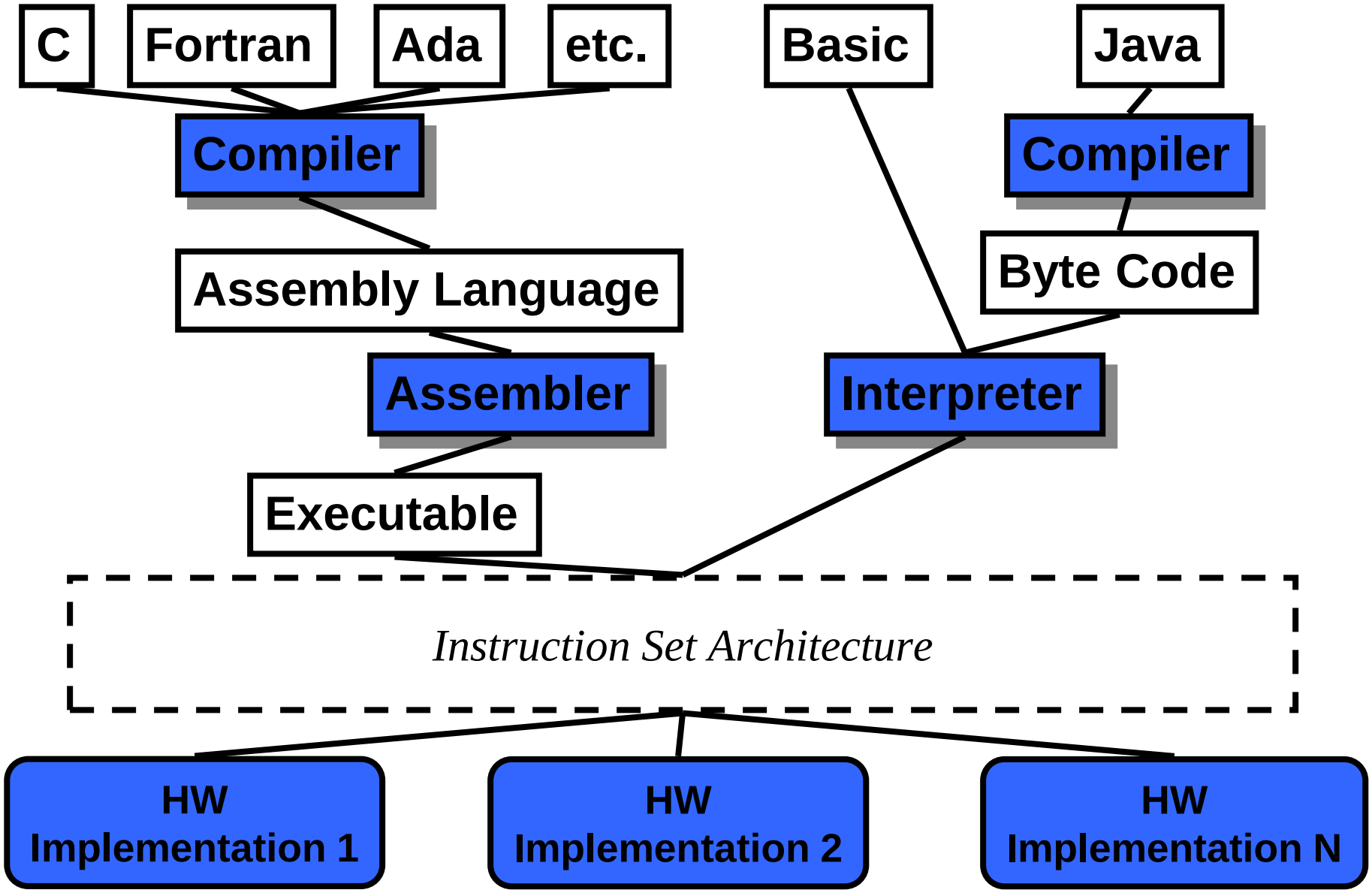
The successive over-relaxation (SOR) : a method for solving a linear system of equations.

The Computer Level Hierarchy

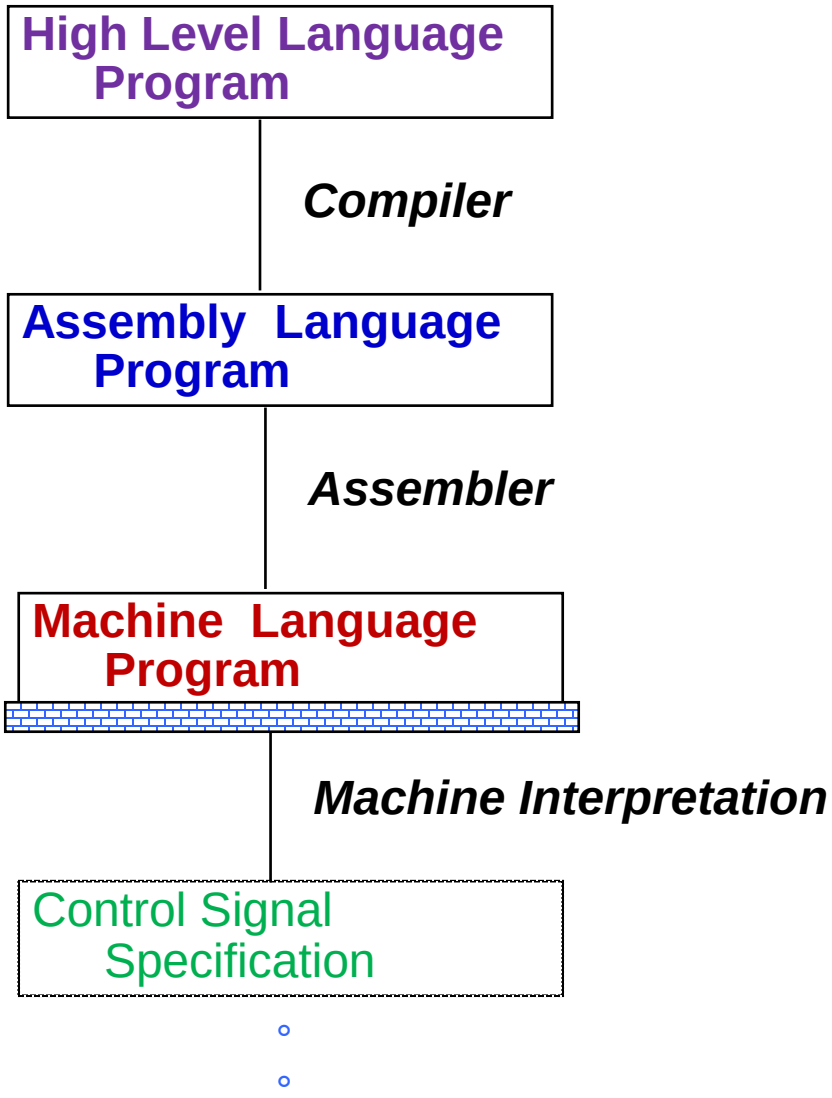
- Each virtual machine layer is an abstraction of the level below it.
- The machines at each level execute their own particular instructions, calling upon machines at lower levels to perform tasks as required.
- Computer circuits ultimately carry out the work.



- Software?
  - –Program or collection of programs.
  - –Enables the hardware to process data.


C
Fortran
Ada
etc.
Basic
Java
Compiler
Compiler
Assembly Language
Byte Code
Assembler
Interpreter
Executable
Instruction Set Architecture
HW Implementation 1
HW Implementation 2
HW Implementation N

Levels of Representation




```
temp = v[k];
v[k] = v[k+1];
v[k+1] = temp;
```

```
lw$15,    0($2)
lw$16,    4($2)
sw        $16,    0($2)
sw        $15,    4($2)
```

```
00001001110001101010111101011000
10101111010110000000100111000110
11000110101011110101100000001001
01011000000010011100011010101111
```

```
ALUOP[0:3] <= InstReg[9:11] & MASK
```

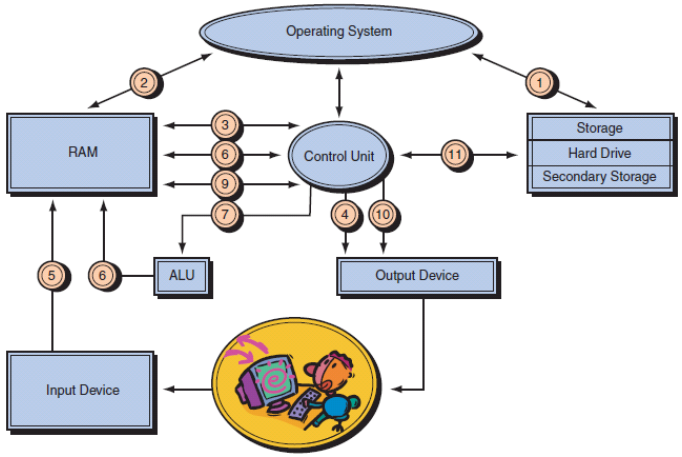
Programming

- Methodologies for creating computer programs that perform a desired function.
  - –Problem Solving
    - How do we figure out what to tell the computer to do?
    - Convert problem statement into algorithm, using stepwise refinement.
    - Convert **algorithm** into machine instructions.
  - –Debugging
    - How do we figure out why it didn't work?
    - Examining registers and memory, setting breakpoints, etc.

Time spent on the first can reduce time spent on the second!

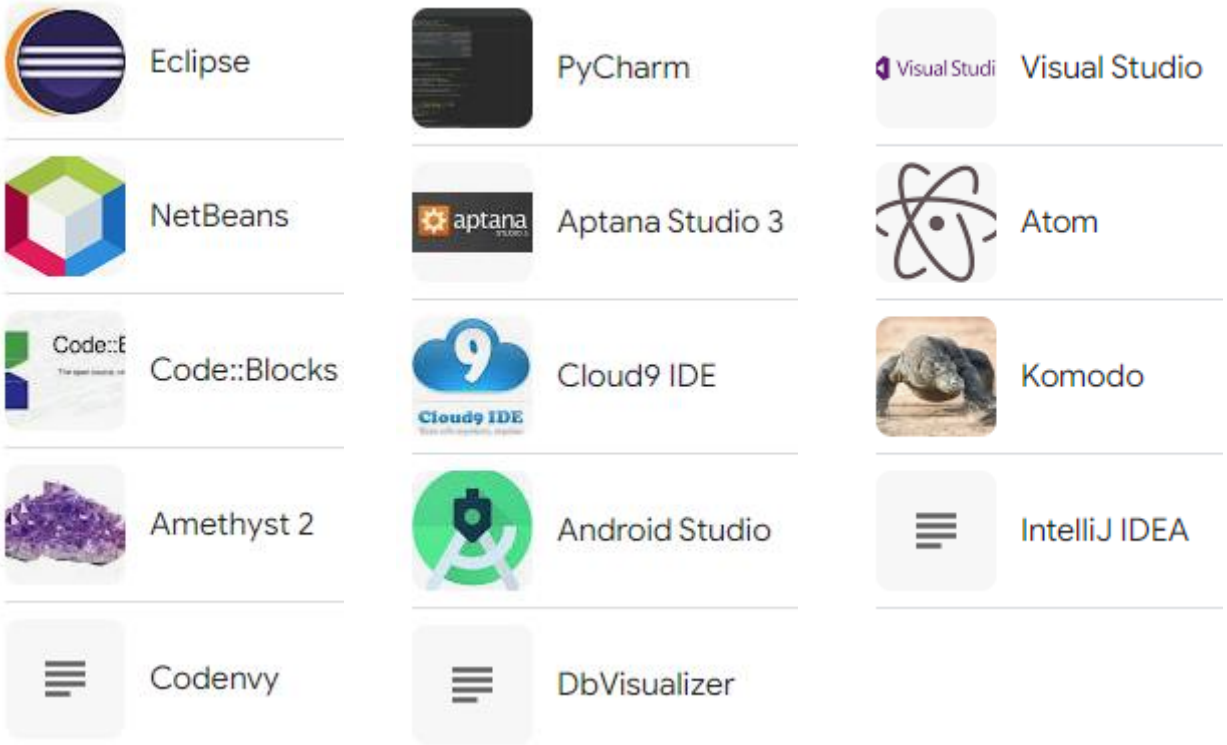
What is a Program?

- Program:
  - A set of instructions to be carried out by a computer.
  - A sequence of steps
    - For each step, an arithmetic or logical operation is done
    - For each operation, a different set of control signals is needed
    - For each operation a unique code is provided
      - e.g. ADD, MOVE
    - A hardware segment accepts the code and issues the control signals
- Program execution:
  - The act of carrying out the instructions contained in a program.
- Programming language:
  - A systematic set of rules used to describe computations in a format that is editable by humans
  - Abbreviated forms of instructions that translate into machine language



Programming Tools Overview

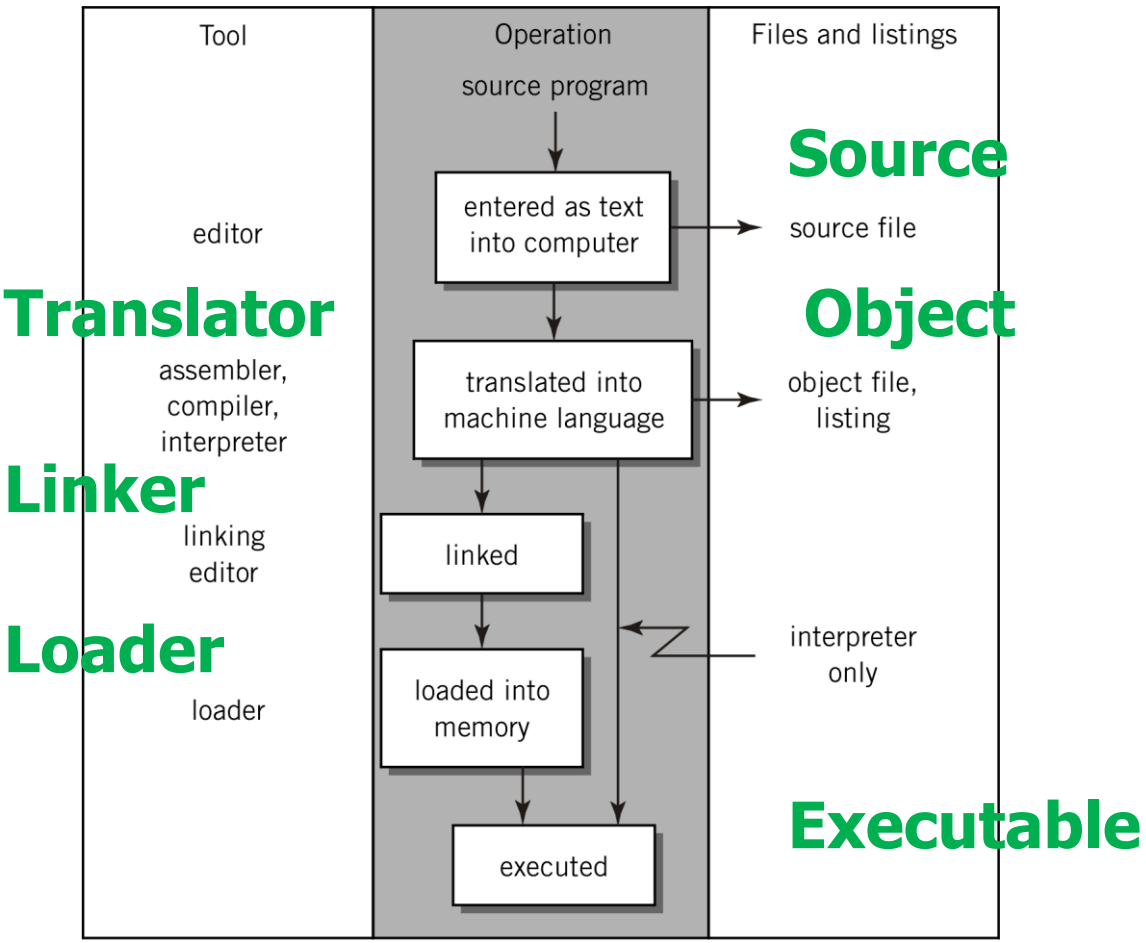
- Editors
- Assemblers
- Debuggers
- Compilers
- Linkers
- Loaders
- Interpreters



- Integrated Development Environments (IDEs) combine several of the above programming tools

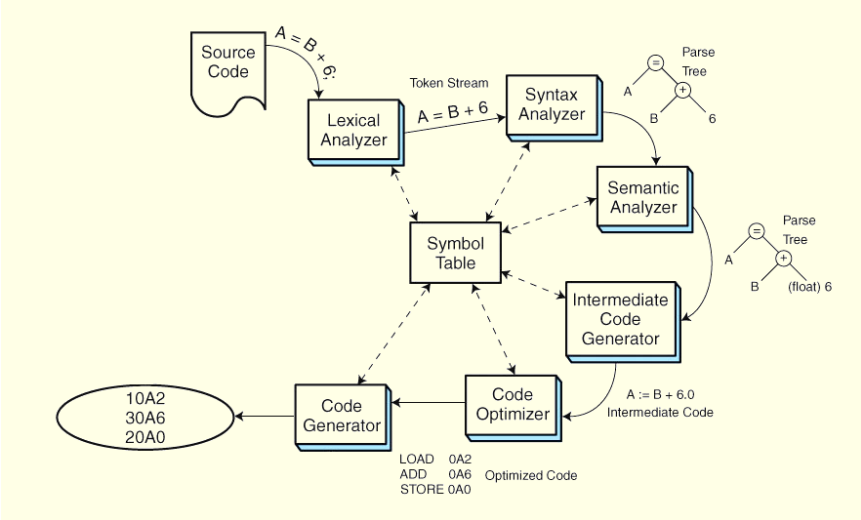
Program Translation Process

- Terms, terms, and more terms!



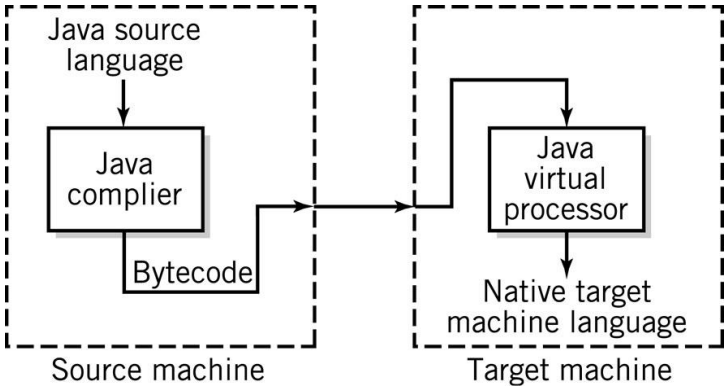
Program Translation Process

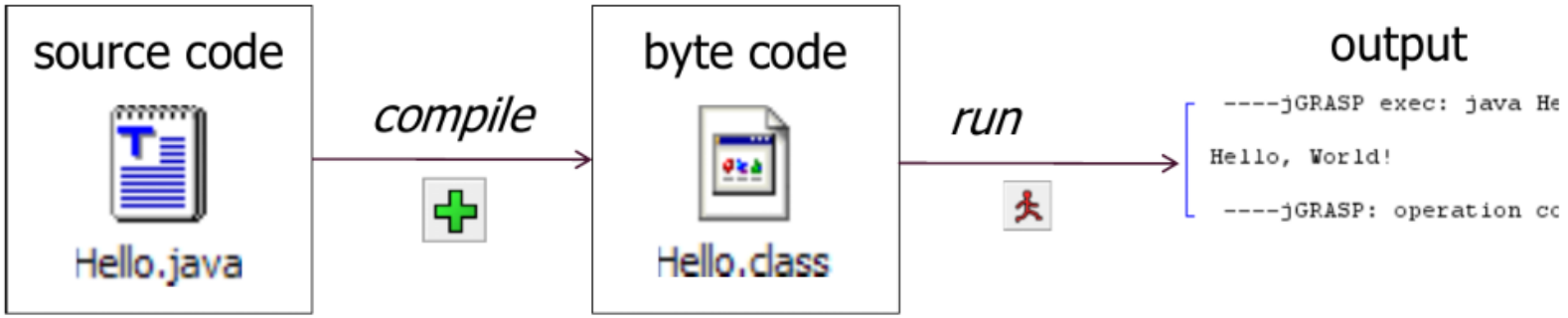
- Program Translation is a six-phase process
  - Lexical analysis extracts tokens, e.g., reserved words and variables.
  - Syntax analysis (parsing) checks statement construction.
  - Semantic analysis checks data types and the validity of operators.
  - Intermediate code generation creates three address code to facilitate optimization and translation.
  - Optimization creates assembly code while taking into account architectural features that can make the code efficient.
  - Code generation creates binary code from the optimized assembly code.



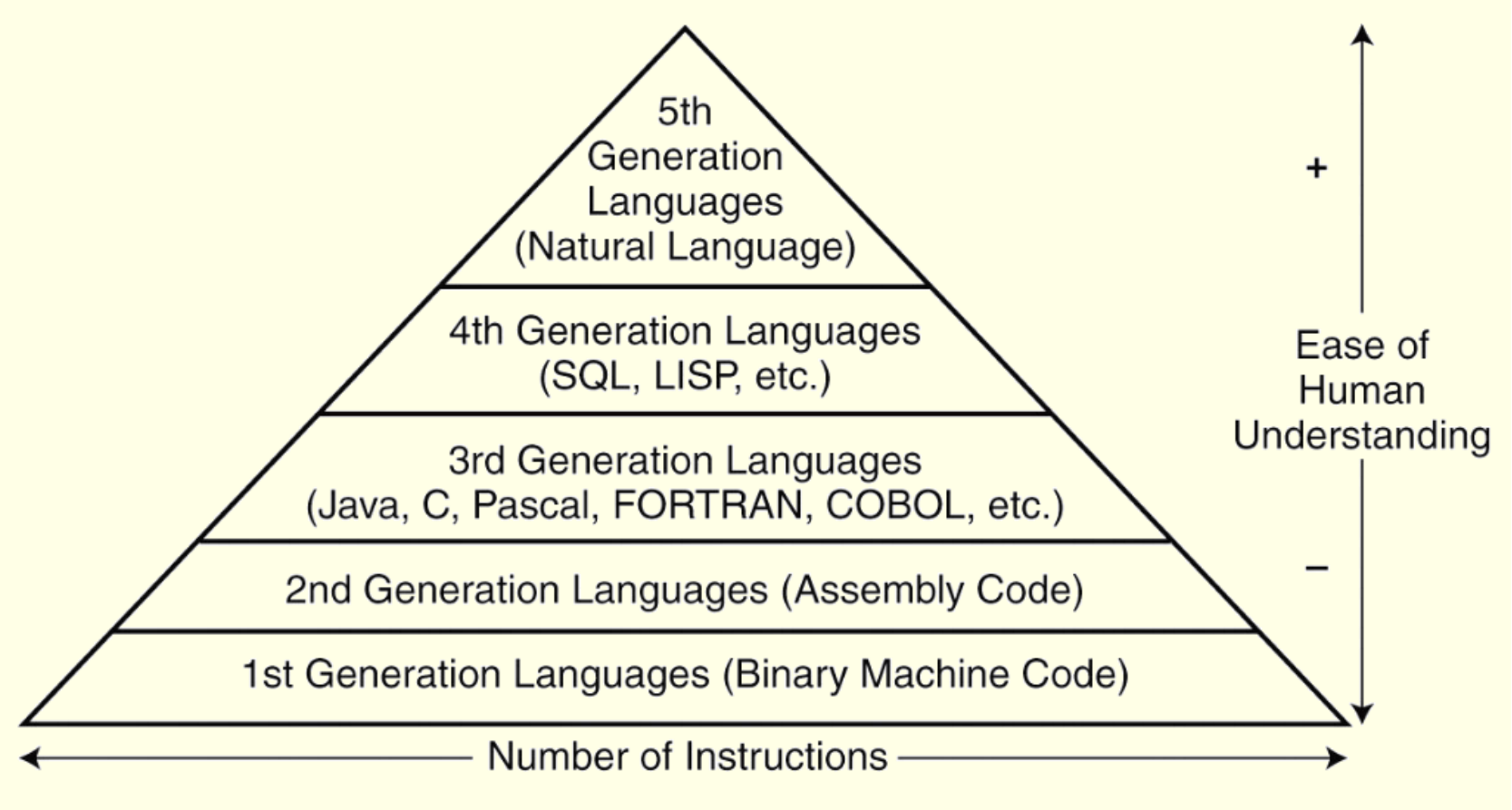
Compile/Run a Program

- **Write** it
  - Code or source code: The set of instructions in a program
- **Compile** it
  - Compile: Translate a program from one language to another
    - Byte code: The Java compiler converts your code into a format named byte code that runs on many computer types
- **Run** (execute) it:
  - Output: The message printed to the user by a program





Evolution of programming languages



- Keep in mind that the computer can understand only the 1GL!

Some Programming Languages

Domain-specific abstraction

Fortran

Cobol

General-purpose abstraction

Simula

Java

Direct mapping to hardware

C++

C++11

C#

Assembler

BCPL

C

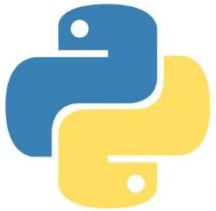
Source: B.Stroustrup, The Essence of C++

Programming Language Categories

- Machine Language
  - Binary coded instructions
- Assembly Language
  - Symbolic coded instructions
- Procedural Languages
  - procedural statements or arithmetic notation
- Fourth-generation Languages
  - Natural language and nonprocedural statements
- Object-oriented Languages
  - Combination of objects and procedures
    - Emphasis on the objects involved in the task, not on the procedure.
    - An object encapsulates a data set with the code that is used to operate on it.
    - Standardized programming modules can be reused.
    - Applications can be rapidly developed with appropriate objects from an object library.

Object-Oriented Languages

- OO Languages have been around since 60s.
  - First example is Simula (first stable release in 1967.)
  - Java
  - C#
  - Python
  - Ruby
  - PHP
  - TypeScript
  - Kotlin
  - R
  - Swift
  - C++
  - Julia
  - Go
  - Perl
  - Smalltalk
  - ...
    - https://en.wikipedia.org/wiki/List_of_object-oriented_programming_languages

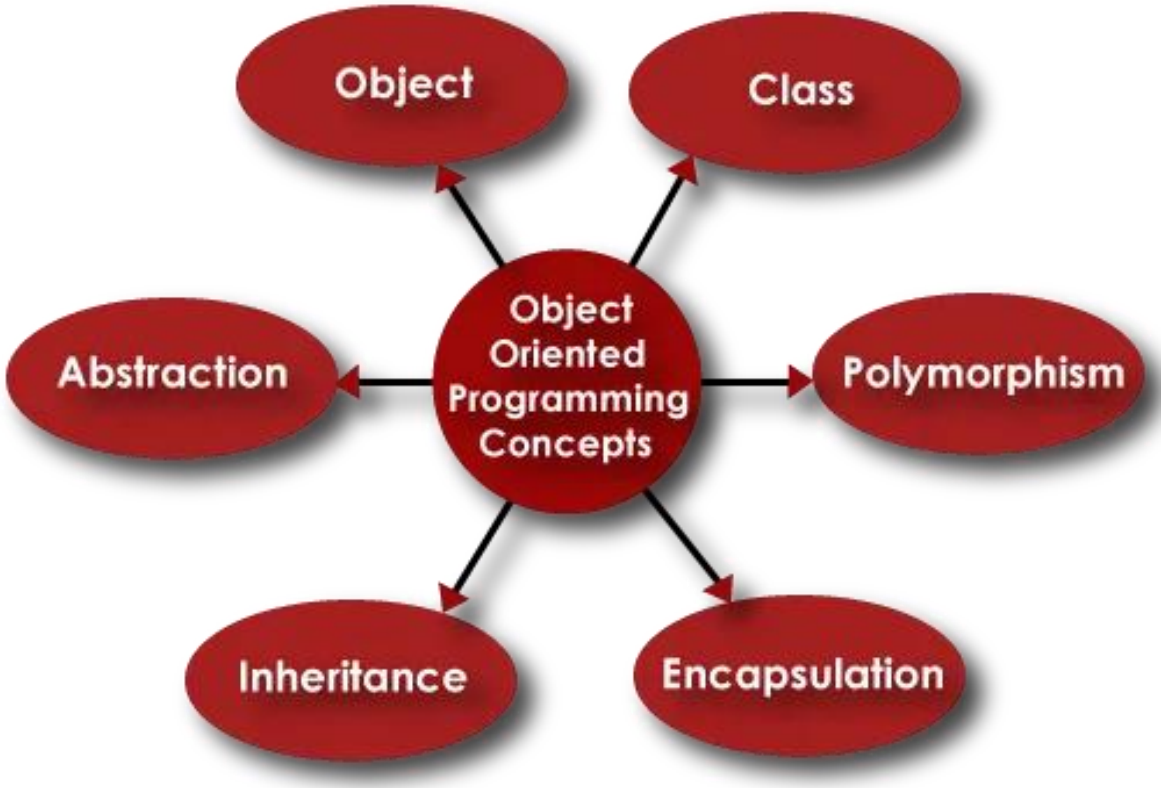


What Is Object-oriented Programming?

- type of computer programming paradigm that centers around the concept of objects.
  - In OOP, an object is a self-contained component that contains data in the form of fields (often known as attributes) and code in the form of methods.
- The essence of OOP
  - to encapsulate elements of a program (such as variables, data structures, and functions) within objects
- increased code reusability
  - by creating different classes for different types of information
- saves development time and makes programs easier to maintain
- suited for designing and building complex computer programs

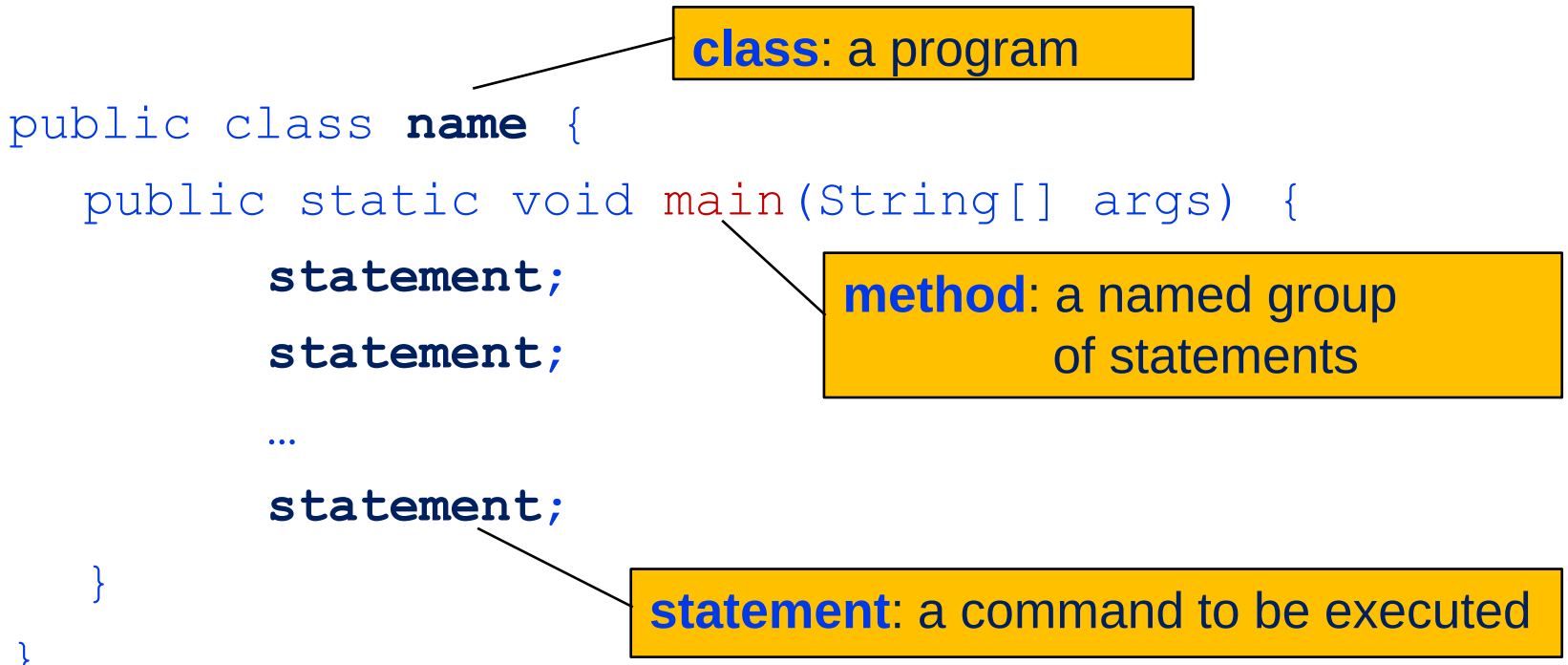
Building Blocks of Object-oriented Programming

- The eight basic concepts of OOP
  - –Classes.
  - –Objects.
  - –Methods.
  - –Attributes.
  - –Inheritance.
  - –Encapsulation.
  - –Abstraction.
  - –Polymorphism



Structure of a Java Program

- Every executable Java program consists of a class,
  - –that contains a method named main,
    - that contains the statements (commands) to be executed.



Names and identifiers

- You must give your program a name.

```
public class GangstaRap {
```

  - Naming convention:
    - Capitalize each word (e.g. `MyClassName`)
  - Your program's file must match exactly (`GangstaRap.java`)
    - Includes capitalization (Java is case-sensitive)

- Identifier:
  - A name given to an item in your program.
- Must start with a letter or _ or $
- Subsequent characters can be any of those or a number
  - legal: `_myName  TheCure  ANSWER_IS_42  $bling$`
  - illegal: `me+u  49ers  side-swipe  Ph.D's`

Keywords

- keyword:
  - An identifier that you cannot use because it already has a reserved meaning in Java

abstract	default	if	private	this
boolean	do	implements	protected	throw
break	double	import	**public**	throws
byte	else	instanceof	return	transient
case	extends	int	short	try
catch	final	interface	**static**	**void**
char	finally	long	strictfp	volatile
class	float	native	super	while
const	for	new	switch	
continue	goto	package	synchronized	

Syntax

- syntax:
  - The set of legal structures and commands that can be used in a particular language
    - Every basic Java statement ends with a semicolon (**;**)
    - The contents of a class or method occur between **{** and **}**

- syntax error (compiler error):
  - A problem in the structure of a program that causes the compiler to fail
    - Missing semicolon
    - Too many or too few { } braces
    - Illegal identifier for class name
    - Class and file names do not match

    …

Comments

- comment:
  - A note written in source code by the programmer to describe or clarify the code
    - Comments are not executed when your program runs
- Syntax:

```
// comment text, on one line
```

or

```
/* comment text; may span multiple lines */
```

- Examples:

```
// This is a one-line comment.


/* This is a very long
   multi-line comment. */
```

Using comments

- Where to place comments:
  - At the top of each file (a "comment header")
  - At the start of every method
  - To explain complex pieces of code

- Comments are useful for:
  - Understanding larger, more complex programs.
  - Multiple programmers working together, who must understand each other's code.

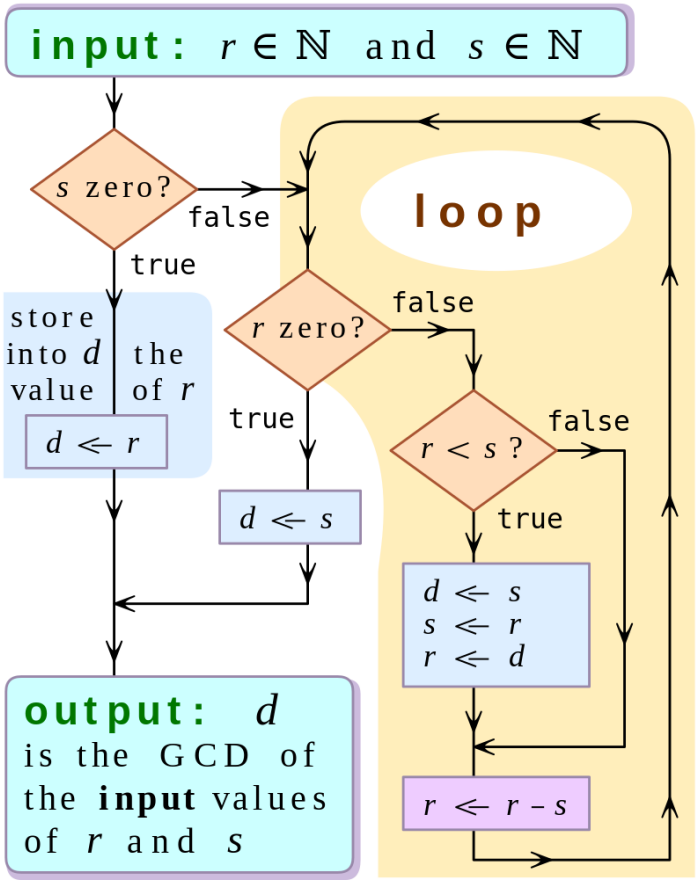
The Concept of an Algorithm

- An algorithm is an ordered set of unambiguous, executable steps that defines a terminating process
  - The steps of an algorithm can be sequenced in different ways
  - Linear (1, 2, 3, …)
  - Parallel (multiple processors)
  - Cause and Effect (circuits)
- A Terminating Process
  - Culminates with a result
  - Can include systems that run continuously
    - Hospital systems
    - Long Division Algorithm
- A Non-terminating Process
  - Does not produce an answer
    - "Non-deterministic Algorithms"

The Concept of Algorithm

- Some Algorithms:
  - Converting from one base to another
  - Correcting errors in data
  - Compression
  - ⋮
  - …
- Many researchers believe that every activity of the human mind is the result of an algorithm
- Etymology:
  - Comes from the Latinization of Al-Khwarizmi's name
    - Around 825 AD, Muḥammad ibn Mūsā al-Khwārizmī wrote kitāb al-ḥisāb al-hindī ("Book of Indian computation") and kitab al-jam' wa'l-tafriq al-ḥisāb al-hindī ("Addition and subtraction in Indian arithmetic").
    - Latin translations:
      Liber **Alghoarismi** de practica arismetrice,
      Liber **Algorismi** de numero Indorum,

Flowchart of using successive subtractions to find the greatest common divisor of number *r* and *s*



The Abstract Nature of Algorithms

- There is a difference between an algorithm and its representation.
  - –Analogy:
    - difference between a story and a book
- A Program
  - –a representation of an algorithm.
- A Process
  - –the activity of executing an algorithm.
- Algorithm Representation
  - –is done formally with well-defined Primitives
    - A collection of primitives constitutes a programming language.
  - –is done informally with Pseudocode
    - Pseudocode is between natural language and a programming language.

Algorithm Discovery

- The first step in developing a program
  - More of an art than a skill
  - A challenging task
- Polya's Problem Solving Steps
  1. Understand the problem.
  2. Devise a plan for solving the problem.
  3. Carry out the plan.
  4. Evaluate the solution for accuracy and its potential as a tool for solving other problems.
- Solve an easier related problem
  - Relax some of the problem constraints
  - Solve pieces of the problem first (bottom-up methodology)
- Stepwise refinement:
  - Divide the problem into smaller problems (top-down methodology)

Algorithm - Example

- Ages of Children Problem:
  - Person A is charged with the task of determining the ages of B's three children.
    - B tells A that the product of the children's ages is 36.
    - A replies that another clue is required.
    - B tells A the sum of the children's ages.
    - A replies that another clue is needed.
    - B tells desired triple must be one whose sum appears at least twice in the table
    - A replies that another clue is needed.
    - B tells A that the oldest child plays the piano.
    - A tells B the ages of the three children.
  - How old are the three children?

a. Triples whose product is 36

(1,1,36)	(1,6,6)
(1,2,18)	(2,2,9)
(1,3,12)	(2,3,6)
(1,4,9)	(3,3,4)

b. Sums of triples from part (a)

1 + 1 + 36 = 38	1 + 6 + 6 = 13
1 + 2 + 18 = 21	2 + 2 + 9 = 13
1 + 3 + 12 = 16	2 + 3 + 6 = 11
1 + 4 + 9 = 14	3 + 3 + 4 = 10

Algorithm - Example

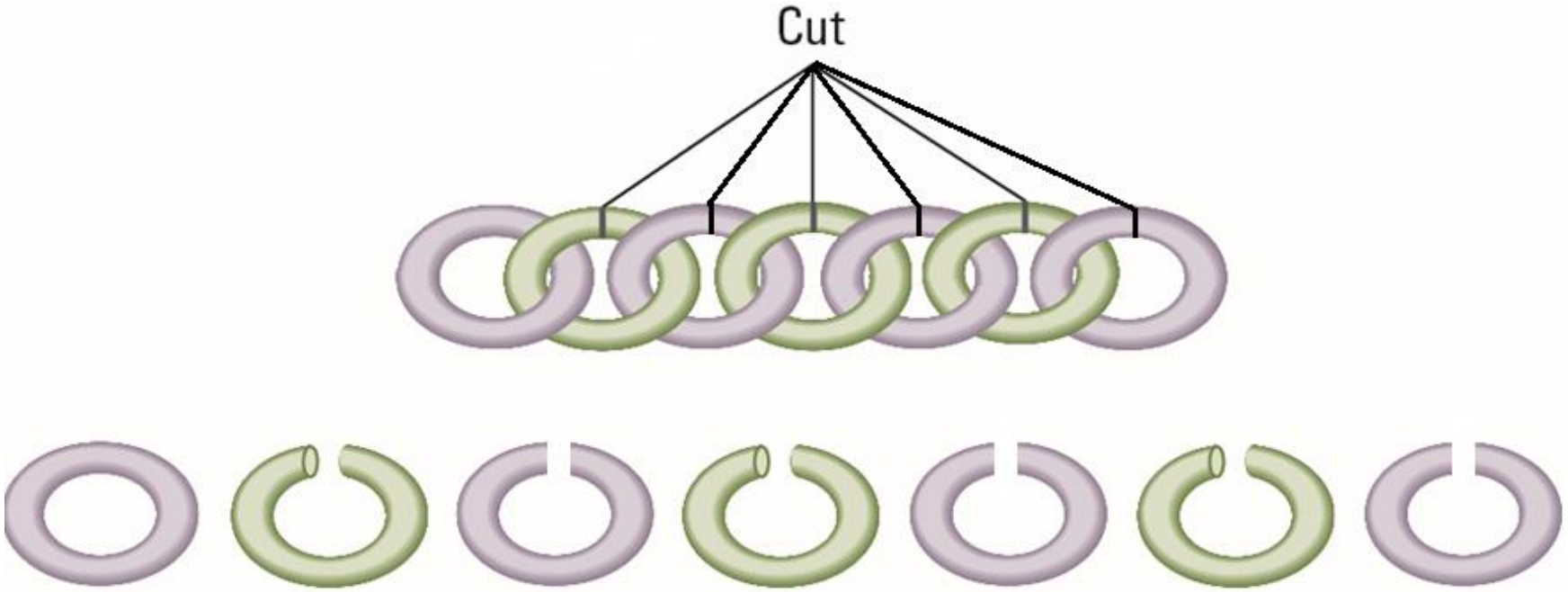
- Chain Separating Problem:
  - –A traveler has a gold chain of seven links.

  - –He must stay at an isolated hotel for seven nights.
  - –The rent each night consists of one link from the chain.
  - –The traveler must pay the hotel one link of the chain each morning

  - –How to pay hotel bill?
    - Constraint
      - –What is the fewest number of links that must be cut so that the traveler can pay the hotel one link of the chain each morning without paying for lodging in advance?

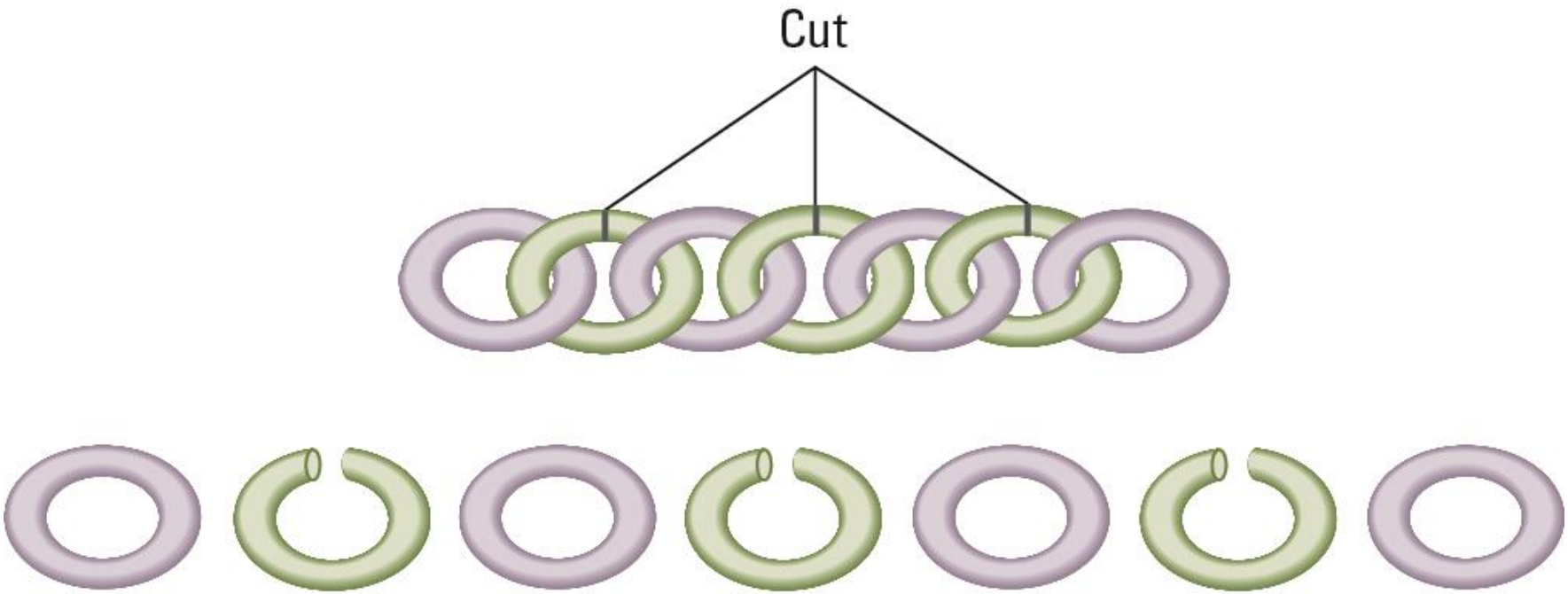
Algorithm - Example

- Separating the chain using six cuts
  - –Cut one link each morning



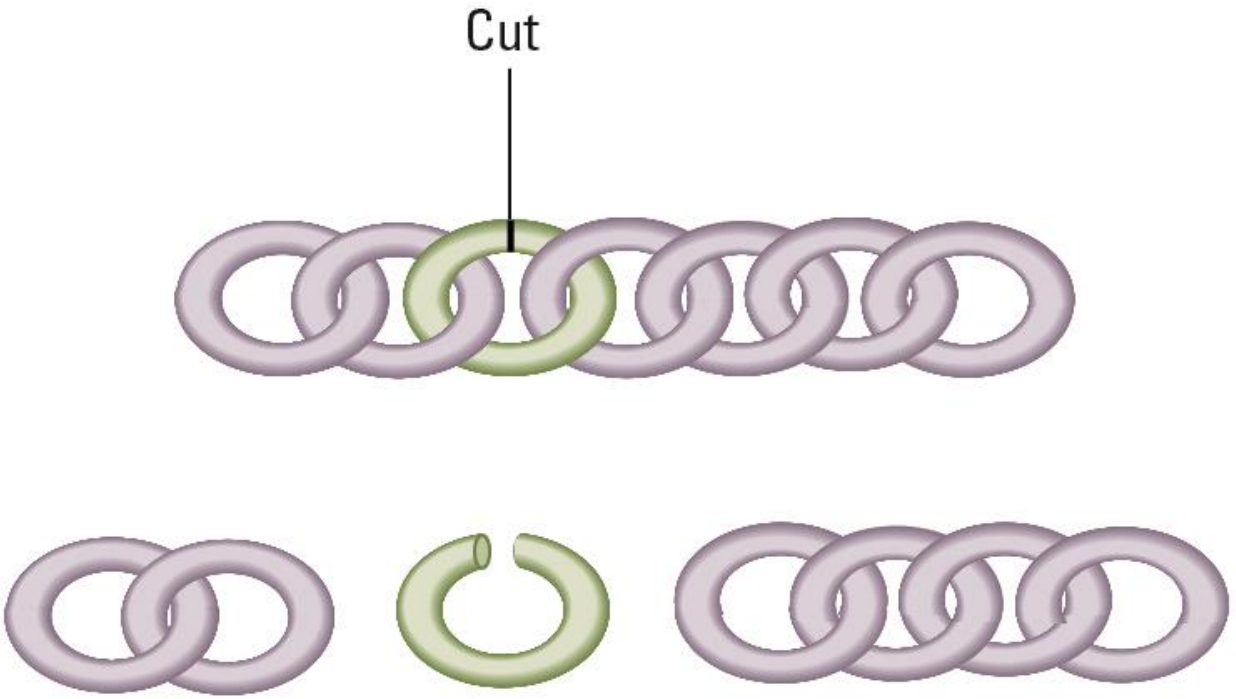
Algorithm - Example

- Separating the chain using only three cuts



Algorithm - Example

- Solving the problem with only one cut



Any Questions?

Hello World Application

Hello World Application

- Develop a Hello World application
  - –We need to implement the HelloWorld class.

```
public class HelloWorld {

}
```

- Access modifier
  - –They are used to set access levels for classes, variables, and other entries.
  - –For the top-level classes, it can be either
    - **public** or
    - **default** (no keyword)
  - –When it is public, this class is visible to the earth and can be accessible from everywhere.

Hello World Application

```
package week1;

class HelloWorld {

}
```

- Without any access modifier this class is visible only within package week1 and can be accessible from package week1 only.
- This default access modifier is also known as package-private.
- **package** is a namespace to organize a set of related classes

Hello World Application

```
private class HelloWorld {

}
```

- A top-level class cannot be private.
  - Illegal modifier for the class HelloWorld

- Access modifiers will be covered in detail later in the course…

Hello World Application

```
public class HelloWorld {

}
```

- The keyword class for the class definition
- The name of the class

Hello World Application

```
package week1;

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

   }
}
```

- No **run** method, instead we have the **main** method
- The entrance point of our program

Hello World Application

```
public static void main(String args[]) {

}
```

- This signature is always the same.

- Main method accepts a single argument
  - An array of element of type String
  - This array holds the command-line arguments

```
java HelloWorld arg1 arg2
```

- Command-line arguments are passed to the program from the command line following the name of the program.
  - They are like parameters of the program.

Hello World Application

```
public static void main(String args[]) {

}
```

- The access modifier
  - It is **public**, therefore can be reached from anywhere.

- This method is **static** means that we can call this method without initializing an object from this class.
  - Since this is the first method being called during the execution of the program, no object has been instantiated yet.

- More details of this will be described later in this course.

Hello World Application

```
public static void main(String args[]) {

}
```

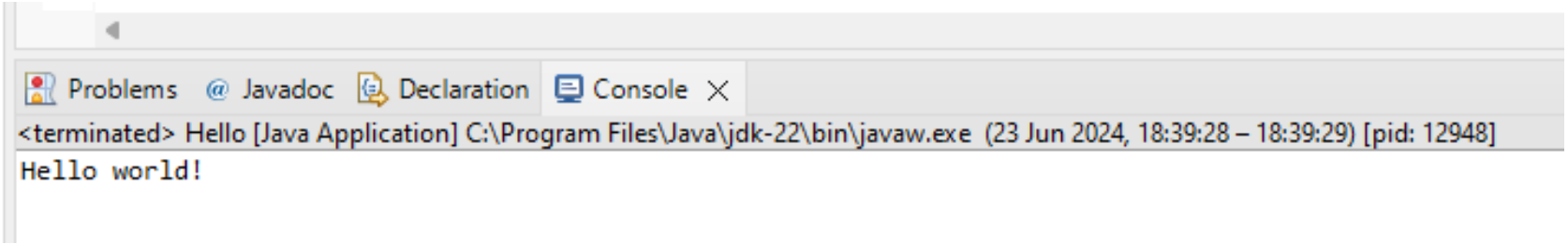
- The return type of the method.

- Nothing to return, since there is not a single line of code waiting for this function's return.
  - However, the main function may return some platform specific values which indicate the final status of the program.

Hello World Application

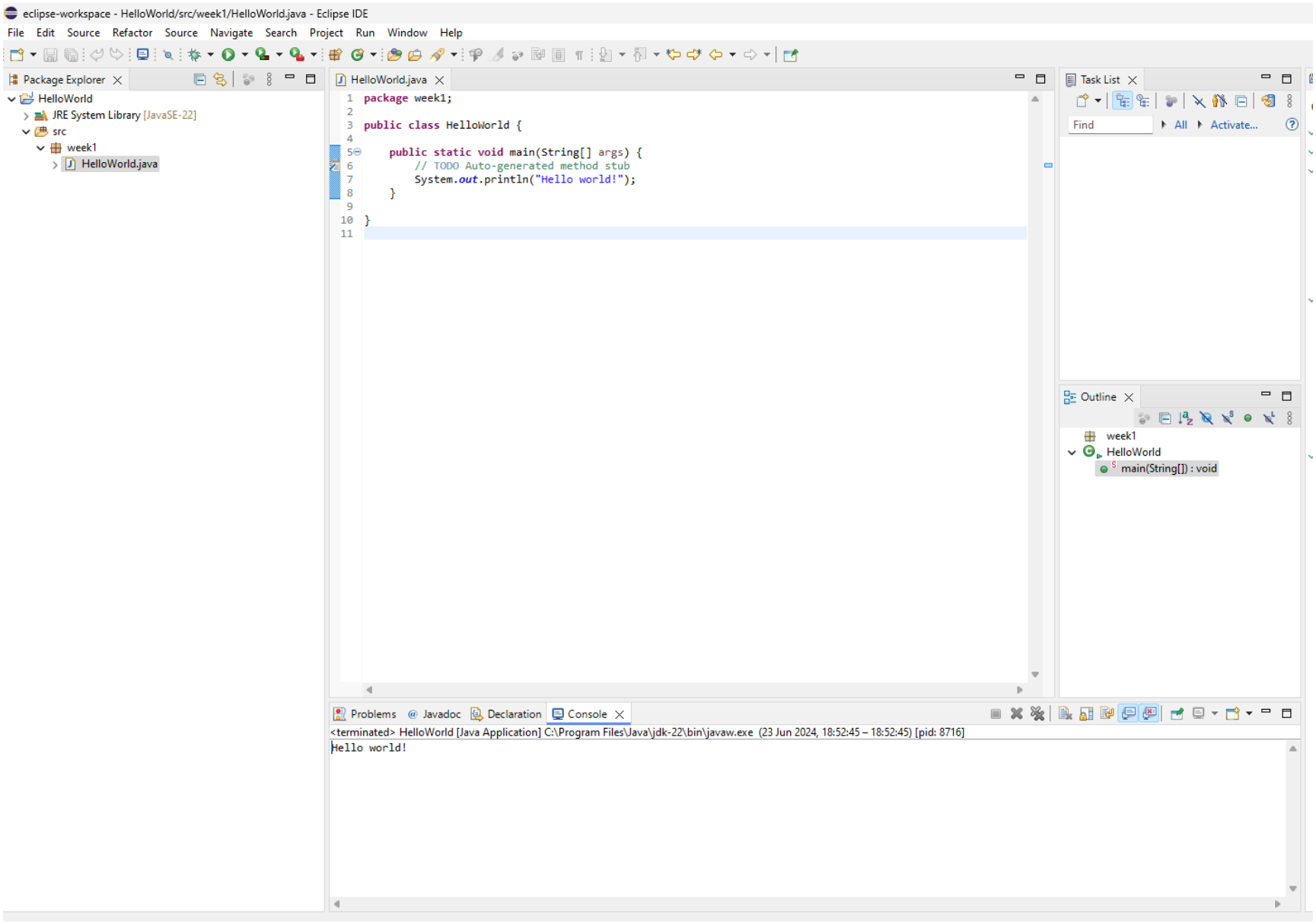
```
package week1;

public class HelloWorld {
   public static void main(String args[]) {
         System.out.println("Hello World"):
   }
}
```

- After running:



- Sending **println** message to the **out** field of the **System** class which is the standard output.

Screenshot of the Eclips IDE



```java
package week1;

public class HelloWorld {

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

Any Questions?