COMP 110 – INTRODUCTION TO PROGRAMMING

Instructor: Prasun Dewan
PREREQUISITES

- None
### Computer World

<table>
<thead>
<tr>
<th>Hardware</th>
<th>Memory</th>
</tr>
</thead>
<tbody>
<tr>
<td>Operating System</td>
<td>Memory Page</td>
</tr>
<tr>
<td>Program</td>
<td>Memory Word</td>
</tr>
<tr>
<td>Processor</td>
<td>Memory Address</td>
</tr>
<tr>
<td>Instruction (e.g. add 2 to 5)</td>
<td>Running a Program</td>
</tr>
<tr>
<td>Source Code</td>
<td>Interactive Program</td>
</tr>
<tr>
<td>Object Code</td>
<td>Non-interactive (Batch) Program</td>
</tr>
<tr>
<td>Programming Language</td>
<td>Program arguments</td>
</tr>
<tr>
<td>Machine Language</td>
<td>Runtime</td>
</tr>
<tr>
<td>Programmers</td>
<td>Editor</td>
</tr>
<tr>
<td>Library (of Code)</td>
<td>Editing Programs</td>
</tr>
<tr>
<td>Translator (Compilers/Interpreter)</td>
<td>Lexical Error</td>
</tr>
<tr>
<td>Users</td>
<td>Syntax Error</td>
</tr>
<tr>
<td>Disks</td>
<td>Semantics Error</td>
</tr>
<tr>
<td></td>
<td>Logic Error</td>
</tr>
<tr>
<td></td>
<td>Debugging</td>
</tr>
<tr>
<td></td>
<td>Style Principles</td>
</tr>
</tbody>
</table>
THEATER ANALOGY

- Play
- Concert
- Talk
- Speech
- Cooking Lessons
- Town Hall Meeting
**Theater World – Part 1**

- Management
  - Manages
- Stage
- Performer
- Assistant
  - Stagehand
- Audience
- Performance Script
  - Written in Performance Language
  - Follows
  - Stored in Notebook or Teleprompter
  - Fetched from Archive
- Real, Trial
THEATER WORLD – PART 2

- Performance Script
  - Written in Performance Language
  - Writes entire script
  - Reads entire script
  - Speaks sentence
  - Hears sentence

- Original Script
  - Written in Script Language

- Translator
  - Writes entire script
  - Reads entire script

- Translator (Interpreter)
  - Speaks sentence
  - Hears sentence
THEATER WORLD – PART 3

Reference Material

Typewriter, Text Editor

Typewriter, Text Editor

Word Processor, Script Writer
Translator / Interpreter, Stagehand, Performer

Checked by

Syntax, Semantics, Logic, & Style

Problems

Written by

Written in

Original Script

Script Language
Computer World – Part 1

- Processor
- Operating System
- Computer
- Assists Runtime
- Interacts with Runtime
- Users
- Real Debugging
- Written in Machine Language
- Stored in Memory
- Fetched from Disk, Tape
- Follows Machine Language

Operating System
Manages

Debugging

Real
**Computer World – Part 2**

- **Source Code**: Written in **Programming Language**
  - **Interpreter** reads statements and writes object code.
  - **Compiler** reads entire source code and writes object code.

- **Object Code**: Written in **Machine Language**
  - **Compiler** writes entire source code.
  - **Interpreter** reads statements.

- **Source Code**
  - **Compiler** reads source code.
  - **Interpreter** reads statements.

- **Object Code**
  - **Compiler** writes entire source code.
  - **Interpreter** writes object code.

**Diagram Notes**:
- **Compiler** and **Interpreter** are associated with the source code and object code, respectively.
- **Programming Language** and **Machine Language** are used to write the code.
TRANSLATING USING BOTH A COMPILER AND AN INTERPRETER

Source Code

Compiler

Platform Independent Intermediate Code

Interpreter

Platform Specific Machine Code

Processor
THEATER WORLD – PART 3

Libraries

Typewriter, Text Editor

Linker

Uses

Editor, Programmer, Translator, Operating System Runtime, Processor

Checked by

Syntax, Semantics, Logic, & Style

Problems

Written by

Written in

Source Code

Programming Language
# Computer vs. Theater

<table>
<thead>
<tr>
<th>Computer</th>
<th>Theater</th>
</tr>
</thead>
<tbody>
<tr>
<td>Hardware</td>
<td>Theater</td>
</tr>
<tr>
<td>Operating System</td>
<td>Theater Management</td>
</tr>
<tr>
<td>Program</td>
<td>Performance</td>
</tr>
<tr>
<td>Processor</td>
<td>Performer</td>
</tr>
<tr>
<td>Instruction (e.g. add 2 to 5)</td>
<td>Performance action (e.g. walk 3 steps.)</td>
</tr>
<tr>
<td>Source Code</td>
<td>Original Script</td>
</tr>
<tr>
<td>Object Code</td>
<td>Performance Script</td>
</tr>
<tr>
<td>Programming Language</td>
<td>Script Language (e.g. German)</td>
</tr>
<tr>
<td>Machine Language</td>
<td>Performance Language (e.g. English)</td>
</tr>
<tr>
<td>Programmers</td>
<td>Script Writers</td>
</tr>
<tr>
<td>Library (of Code)</td>
<td>Reference Material (from Books)</td>
</tr>
<tr>
<td>Translator (Compilers/Interpreter)</td>
<td>Translator (Before/During Performance)</td>
</tr>
<tr>
<td>Users</td>
<td>Audience</td>
</tr>
<tr>
<td>Disks</td>
<td>Archival Storage Areas</td>
</tr>
<tr>
<td>Memory</td>
<td>Script performance notebook accessible to performers</td>
</tr>
<tr>
<td>Hardware</td>
<td>Theater</td>
</tr>
</tbody>
</table>
# Computer vs. Theater

<table>
<thead>
<tr>
<th>Computer</th>
<th>Theater</th>
</tr>
</thead>
<tbody>
<tr>
<td>Memory Page</td>
<td>A Notebook Page</td>
</tr>
<tr>
<td>Memory Word</td>
<td>A Notebook Line</td>
</tr>
<tr>
<td>Memory Address (Page Number, Word Number)</td>
<td>Line Identification (Page Number, Line Number)</td>
</tr>
<tr>
<td>Running a Program</td>
<td>Performing a Script</td>
</tr>
<tr>
<td>Interactive Program</td>
<td>Performance with audience participation</td>
</tr>
<tr>
<td>Non-interactive (Batch) Program</td>
<td>Performance with no audience participation</td>
</tr>
<tr>
<td>Program arguments</td>
<td>Special instructions at start of performance</td>
</tr>
<tr>
<td>Runtime</td>
<td>Stage-Hands</td>
</tr>
<tr>
<td>Editor</td>
<td>Typewriter/Wordprocessor</td>
</tr>
<tr>
<td>Editing Programs</td>
<td>Writing Scripts</td>
</tr>
<tr>
<td>Lexical Error</td>
<td>Spelling Error</td>
</tr>
<tr>
<td>Syntax Error</td>
<td>Grammar Error</td>
</tr>
<tr>
<td>Semantics Error</td>
<td>Inconsistencies in Script</td>
</tr>
<tr>
<td>Logic Error</td>
<td>Execution Error</td>
</tr>
<tr>
<td>Debugging</td>
<td>Staging trial performances</td>
</tr>
<tr>
<td>Style Principles</td>
<td>Style Principles</td>
</tr>
</tbody>
</table>
COMPUTER VS. THEATER

- CPU is fast: can do several performances at one time.
- CPU is dumb: no improvisation possible.
- Machine language much lower-level than programming language.
Why Java

- Modern, modular (object-oriented) language.
- Good Error Detection.
- Rich Library Embodying Many Good Programming Principles
- Can Write Teaching Tool (ObjectEditor)
JAVA VERSIONS, DIALECTS & PROGRAMMING ENVIRONMENT

- JDK 1.5+
- Eclipse IDE Programming Environment
- Will not matter
COMPUTER VS. PROGRAM MODEL

Processor

Compiler

Program (source code)
STRUCTURING IN SCRIPTS

Script

Follows
STRUCTURING IN SCRIPTS

Script components are abstract.

So are program components!
COMPUTER VS. PROGRAM MODEL

Processor

Compiler

Program (source code)
OUTLINE

- Intuitive Explanation
- Two Concrete Examples
- Calculators
  - square
  - BMI
- Basic Java program elements, programming styles, error handling
- Running and interacting with a Java program
STRUCTURING IN SCRIPTS
SCRIPTING IN SCRIPTS

Script

Introduction

Body

Conclusion

Paragraph 1

Paragraph 2

Sentence 1

Sentence 2

Script components are abstract.

So are program components!

Next chapter discusses abstract program components.