Introduction

**EECS1022**

**MOBILE COMPUTING**

**INTRODUCTION**
*(SLIDES ADAPTED FROM PROF. H. ROUMAN)*

**PROF. Y. LESPÉRANCE**
Dept. of Electrical Engineering & Computer Science

---

**TOPICS & TECHNOLOGIES**

- Abstraction & Separation of Concerns
- The Software Development Cycle
- Object Oriented Programming [OOP]
- Data structures and Algorithms
- Android App Development
- User Interface [UI] Design
- The Java Programming Language

---

**APPROACH**

- Builds on EECS1012
  Separation of Concerns, Computational thinking.

- **Industrial-Strength Tools**
  UI via XML (not HTML), Behavior via Java (not JS).

- **Solid Platform**
  O/S is Android, IDE is Studio.

- **Experiential Pedagogy**
  Foundational concepts in class + real-life projects in lab

---

**ANDROID: THE STACK**

---

EECS1022/Lesperance
**Introduction**

**Android Versions [A/B in 2008]**

**XML**

- Like a Generalized HTML
  But 100% strict.
- Well-Formed XML
  Obey the syntax rules:
  See: [http://www.w3schools.com/xml/xml_syntax.asp](http://www.w3schools.com/xml/xml_syntax.asp)
- Valid XML
  Must be well-formed and obeys a schema that dictates the names of tags and attributes (namespace) and sets the types of their values.

**XML Example — An Android Layout**

```xml
<LinearLayout layout_width="match_parent" orientation="vertical">
  <EditText layout_width="match_parent" id="width"/>
  <EditText layout_width="match_parent" id="height"/>
  <Button layout_width="match_parent" text="Compute" id="button"/>
</LinearLayout>
```

Locate:
- Document root, tag, closing tag, attribute, attribute value

Note the naming style for multi-word identifiers:
- Pascal, Camel, or underscore.

**Java**

- Adopts the C Syntax
  Same as JavaScript
- Strongly-Typed
  Syntax errors exposed as you type. Static checking of potential runtime and logic errors.
- OOP
  Programming by Delegation.
- Platform-Independent
  Write once, run anywhere.
### JAVA EXAMPLE - A CLASS

**public class Rectangle**

```java
private int width;
private int height;
public Rectangle(int w, int h)
{
    this.width = w;
    this.height = h;
}
public int getArea()
{
    int result = this.width * this.height;
    return result;
}
```

### USING LIBRARY CLASSES

- Can use standard Java library classes
- Browse API to find out how to use
- Import the classes you use
- Can use non-standard Java libraries by installing and linking them to your project

### ANDROID STUDIO

*Makes writing code easier (compile-as-you-type); designing UI easier (drag widgets, set properties); running and debugging.*

- **Launch Studio**
  From the Application, Programming menu.

- **Start a New Project**
  Programming by Delegation.

- **Project Location**
  `/home/user/AndroidStudioProjects/`

### THE STUDIO ECOSYSTEM
Introduction

**THE STUDIO ECOSYSTEM**

**COMPUTER SCIENCE**

- Is concerned with information and its processing
- What problems are solvable by algorithms?
- Computational complexity of algorithms and problems
- Programming and abstraction
SEARCH FOR A GIVEN N

List:

<p>| | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>26</td>
<td>30</td>
<td>12</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>17</td>
<td>7</td>
<td>5</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

*Complexity: O(N)*
Search for a Given N

Sorted List:

<p>| | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>7</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>12</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>17</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>26</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>30</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

*Complexity: $O(\log N)$*
Tree:

```
SEARCH FOR A GIVEN N

Complexity: O(lgN)
```
SEARCH FOR A GIVEN N

Hash Table:

|   | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 | 21 | 22 | 23 | 24 | 25 | 26 | 27 | 28 | 29 | 30 | 31 |
|---|---|---|---|---|---|---|---|---|---|---|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|
| 0 | 0 | 0 | 1 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 |

**Complexity: O(1)**