CSE240 – Introduction to Programming Languages (online)

Lecture 05:
Programming with C

Javier Gonzalez-Sanchez

javiergs@asu.edu
javiergs.engineering.asu.edu
Office Hours: By appointment

jgs

Imperative Paradigm

jgs

Fully specified and **fully controlled** **manipulation** of named **data** in a stepwise fashion.

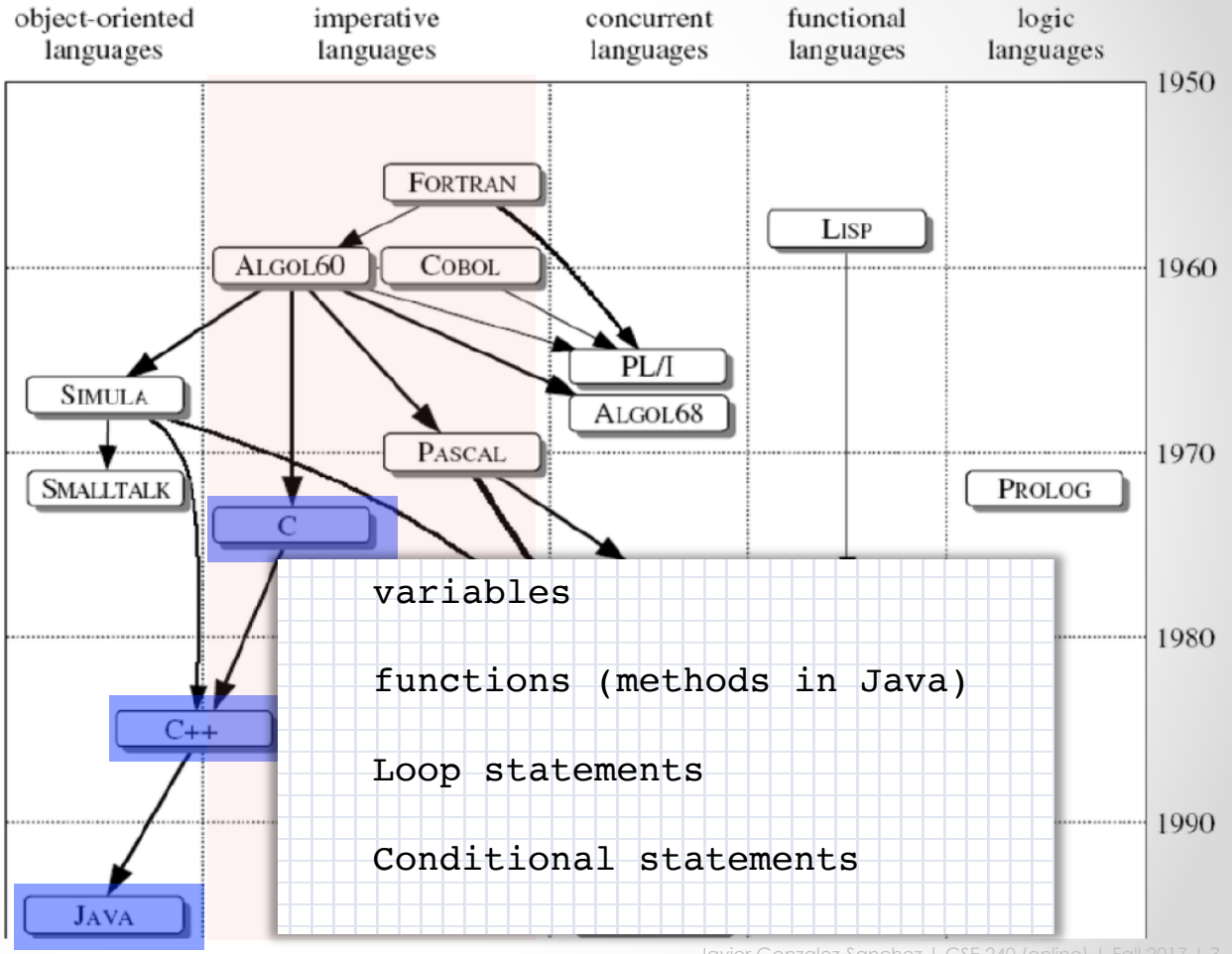
which means that:

Programs are **algorithmic** in nature:
do this, then that, then repeat this ten times

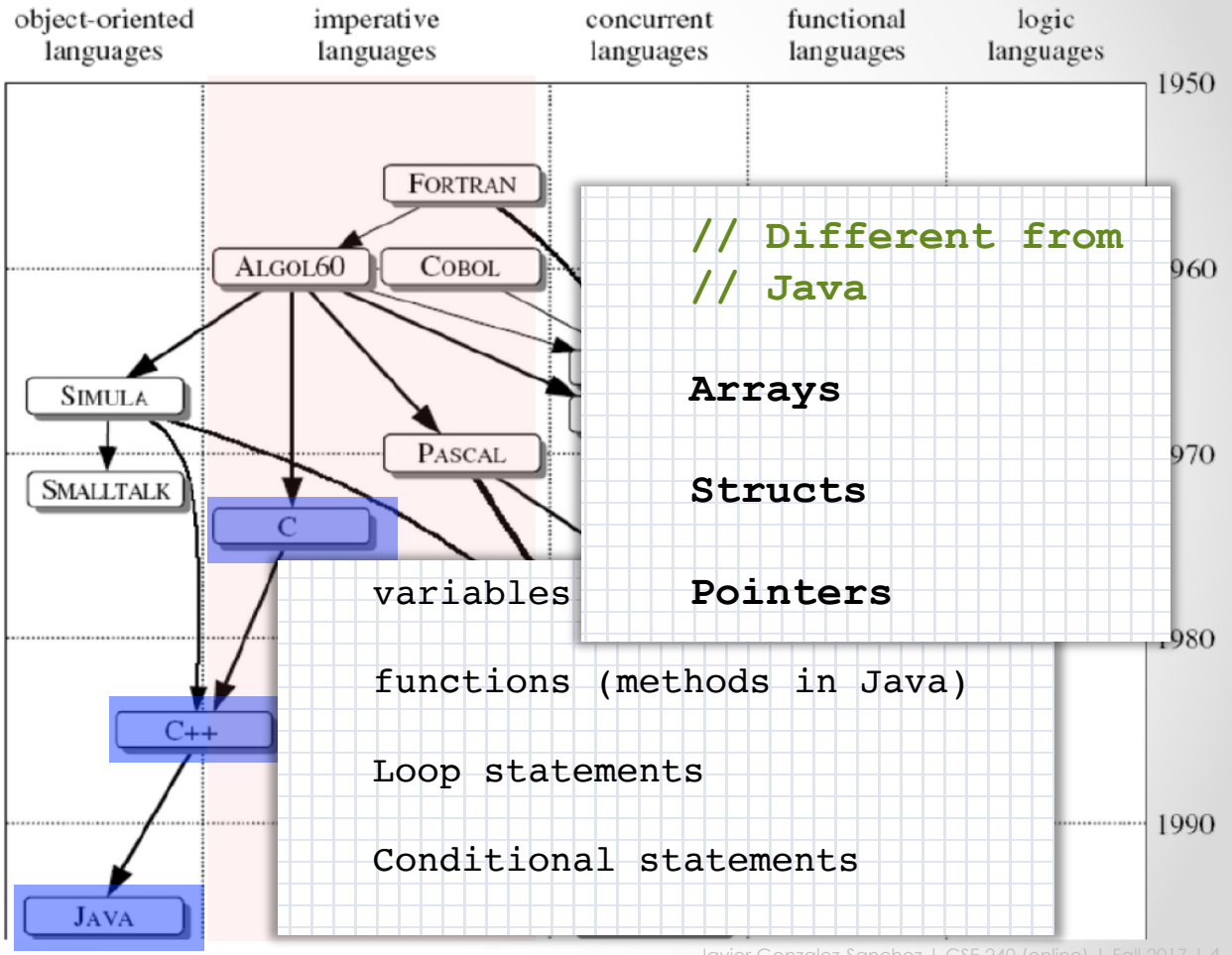
Focuses on how

```
variables

functions (methods in Java)

Loop statements

Conditional statements
```

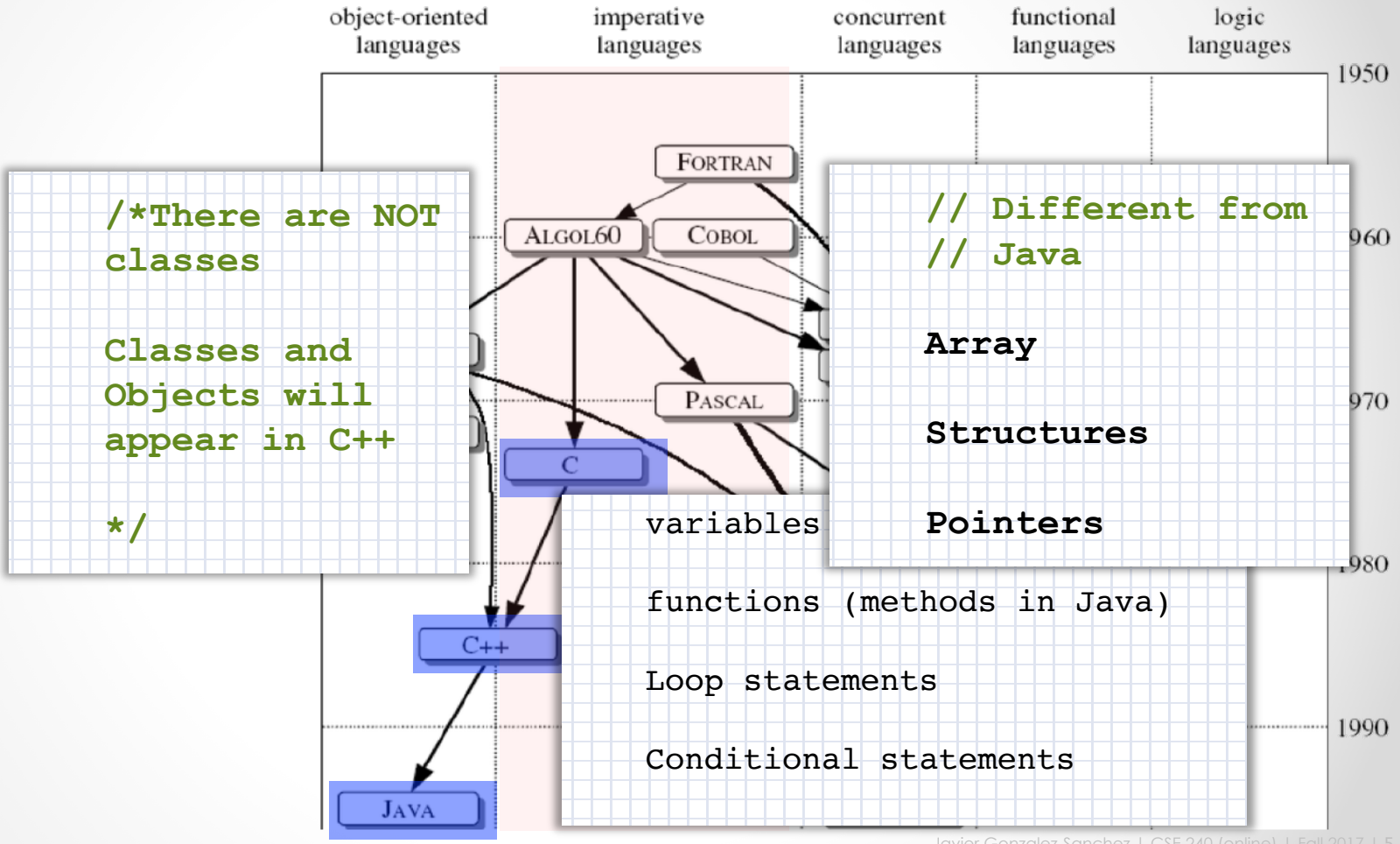
Paradigms


object-oriented languages
imperative languages
concurrent languages
functional languages
logic languages
1950
1960
1970
1980
1990
FORTRAN
LISP
ALGOL60
COBOL
PL/I
SIMULA
ALGOL68
PASCAL
SMALLTALK
PROLOG
C
C++
JAVA
variables
functions (methods in Java)
Loop statements
Conditional statements

Paradigms


object-oriented languages
imperative languages
concurrent languages
functional languages
logic languages
1950
1960
1970
1980
1990
FORTRAN
ALGOL60
COBOL
SIMULA
PASCAL
SMALLTALK
C
C++
JAVA
// Different from
// Java
Arrays
Structs
Pointers
variables
functions (methods in Java)
Loop statements
Conditional statements

Paradigms




```
/*There are NOT
classes

Classes and
Objects will
appear in C++

*/
```

```
// Different from
// Java

Array

Structures

Pointers
```

```
variables

functions (methods in Java)

Loop statements

Conditional statements
```

Outline

1. **Getting started** with Language C

2. Primitive **data types**, **arrays** (and **strings)**

3. **Pointers**

4. **`typedef`**, **`enum`** and **`struct`** type

5. **Functions** calls and **parameter** passing

1. Getting Started

We Need a Compiler

- **Microsoft Visual Studio for C/C++**
  View instructions on Blackboard

- **Online Compiler for C/C++**
  https://www.onlinegdb.com/online_c_compiler

- **Dev-C++ 5**
  http://www.bloodshed.net/devcpp.html

Reference

- **Textbook**
  Section 2.1 to 2.5 and 2.7
  Optional 2.6 (files), 2.7 (recursion). – not included in the exam

- **The Programming Language C by Kernighan and Ritchie**

  https://archive.org/details/TheCProgrammingLanguageFirstEdition

- **Learning C programming**

  https://www.tutorialspoint.com/cprogramming/cprogramming_tutorial.pdf

Anatomy of a Program in C

- The parts (components) of a C program are called **functions**.
- There are built-in functions that exist in **libraries** and user defined functions that are written by the programmers.
- You can declare variables inside and outside functions. They are called **local variables and global variables**, respectively.
- And, a program in C must contain exactly one **main( )** function.

Getting Started


```c
/*
This is an Example
*/

#include <stdio.h>
#include <math.h>

int foo = 0;

// this is a one line comment
int myFunction() {
    return 5;
}

int main() {
    printf("Hello World");
    printf("\n");
    int x = myFunction();
    foo = x + 1;
    printf("Result    : %f", sqrt(2));
    return 0;
}
```

```
Hello World
Result    : 1.414214
```

1. Comments are identical in Java
2. #include is equivalent to import
3. Libraries are *.h files
4. There are not classes
5. Methods are called functions.
6. Global variables, local variables, and parameters.
7. Most of the lexical, syntactical, and semantical rules are as you know from your Java courses

main()


```c
/*
This is an Example
*/

#include <stdio.h>
#include <math.h>

int foo = 0;

// this is a one line comment
int myFunction() {
    return 5;
}

int main() {
    printf("Hello World");
    printf("\n");
    int x = myFunction();
    foo = x + 1;
    printf("Result    : %f", sqrt(2));
    return 0;
}
```

```
Hello World
Result    : 1.414214
```

8. Like Java, main( ) is the entry point for program execution.

   But, (1) C allows void or int as type for main; (2) C allows empty parameters or two parameters for main. This are correct:

- `void main () { }`
- `void main (int argc, char *argv [ ]) { }`
- `int main() {return 0;}`
- `main() {return 0;}` // if there is not a type, C apply as default int
- `int main (int argc, char *argv [ ]) { }`

main()

```
/*
This is an Example
*/
```

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

}
```

```
    return 5;
}

int main() {
    printf("Hello World");
    printf("\n");
    int x = myFunction();
    foo = x + 1;
    printf("Result    : %f", sqrt(2));
    return 0;
}
```

```
Hello World
Result    : 1.414214
```

8. Like Java, main( ) is the entry point for program execution.

...ows void or int as ... (2) C allows empty ...two parameters for ...e correct:

- void main () {
  }
- void main (int argc, char *argv [ ]) { }
- int main() {return 0;}
- main() {return 0;} // if there is not a type, C apply as default int
- int main (int argc, char *argv [ ]) { }

Output


```c
/*
This is an Example
*/

#include <stdio.h>
#include <math.h>

int foo = 0;

// this is a one line comment
int myFunction() {
    return 5;
}

int main() {
    printf("Hello World");
    printf("\n");
    int x = myFunction();
    foo = x + 1;
    printf("Result    : %f", sqrt(2));
    return 0;
}
```

```
Hello World
Result    : 1.414214
```

`printf (control sequence, expressions);`

- The control sequence includes a constant string to be printed, e.g., "Result: ", and control symbols to be used to convert variables from their numeric values that are stored in the computer to printing format.

- The expressions is the list of expressions whose values are to be printed out. Each expression is separated by a comma. This is optional.

Output


```c
/*
This is an Example
*/

#include <stdio.h>
#include <math.h>

int foo = 0;

// this is a one line comment
int myFunction() {
    return 5;
}

int main() {
    printf("Hello World");
    printf("\n");
    int x = myFunction();
    foo = x + 1;
    printf("Result    : %f", sqrt(2));
    return 0;
}
```

```
Hello World
Result    : 1.414214
```

```
printf (control sequence, expressions);
```

- The control sequence includes a constant string to be printed, e.g., "Result: ", and control symbols to be used to convert variables from their numeric values that are stored in the computer to printing format.

%d	for integer
%f	for floating point
%c	for character
%s	for string of characters

Output

```
printf (control sequence, expressions);
```

```java
// Java
int x = 5;
float y = 10.3f;
System.out.println("hello " + x + " bye " + y);

// C
int x = 5;
float y = 10.3;
printf("hello %d bye %f", x, y);
```

Input


```c
#include <stdio.h>

void main () {
  int i, n = 5;
  printf("Hi, please enter an integer: ");

   // input: scanf (control sequence, &variable1, ... &variablen);
   // &variable: address of the variable.
  scanf("%d", &i); // input function

  if (i > n)
    n = n + i;
  else
    n = n - i;
  printf("i = %d, n = %d\n", i, n); //output function
}
```

```
Hi, please enter an integer: 4
i = 4, n = 1
```

Control Statements

jgs

These that you know, work in the same way

- for, while, do/while
- if/else, switch

But, **there are not boolean values**. **Zero is false** and any other number is **true**.

The following program is **correct** and print **"Hello"**

```
1  #include <stdio.h>
2
3  void main () {
4  if (2+2)
5    printf("Hello");
6  else
7    printf("Bye");
8
9  }
```

CSE240 – Introduction to Programming Languages (online)

Javier Gonzalez-Sanchez
javiergs@asu.edu
Fall 2017