Cairo University, Faculty of Computers and AI

CS213 – 2022 / 2023

Object Oriented Programming

Lecture 1: Welcome and Introduction

By

Dr. Mohammad El-Ramly

http://www.acadox.com/class/64401 PY7OGJ

Lecture Objectives

1. Introduction
2. Course Objectives
3. Course Administration
4. Programming Paradigms
5. C++ Revision
6. Readings

جدد النية

- إنما الأعمال بالنيات و إنما لكل امرئ ما نوى
- «يرفع الله الذين آمنوا منكم و الذين أوتوا العلم درجات»

- من سلك طريقا يلتمس فيه علما سهل الله له طريقا إلى الجنة
- العلم يرفع بيتاً لا عماد لــه .... والجهل يهدم بيت العز والكرم

I am ……

- Mohammad El-Ramly
- Assistant Professor of Computer Sciences
- Specialization: Software Engineering
- B.Sc. of Computer Engineering, Ain Shams University, Cairo.
- M.Sc. of Operations Research, Cairo University.
- Ph.D. of Computer Science, University of Alberta, Canada.

I. What is this course about?

- Study in FCAI goes in streams or lines: SE, Networks, Database, Algorithms, …..
- Software Engineering line includes:
  - Fundamentals of CS
  - Structured Programming
  - Object Oriented Programming
  - Data Structures / Database
  - Introduction to Software Engineering
  - Advanced Software Engineering

What is this course about?

- Advance your general programming skills
- Learn advanced OO and C++ techniques
- Topics
  - Review of C++ basics 1 w
  - Object-oriented programming 4 w
    - Basics of OO: objects, classes and data abstraction
    - OO program design and modeling
    - Inheritance, abstraction, polymorphism, overloading,
  - Generic programming in C++ (templates) 1 w
  - Recursion and backtracking 1 w
  - Exception handling 1 w
  - Standard template Library (STL) 1 w
  - Advanced concepts 2 w

What is this course about?

- Review of C++
- OOP C++
  - Function and operator overloading
  - Defining classes
  - Constructors and deconstructors
  - OOP concepts
  - OOP modeling
  - Inheritance and polymorphism
  - Friend functions
  - Abstract class and virtual functions
- Separate compilation
- Recursion and backtracking
- Generic programming in C++
- STL / Exceptions

Why OOP in C++ ?

- C++ programming is pervasive in many **key areas of the software industry**:
  - Computer games and entertainment industry
  - Operating systems
  - Audio/Video processing
  - Computer device drivers
  - Control systems
  - Telecommunications systems
  - Embedded software systems
  - Simulation systems
  - Medical imaging

Why OOP in C++ ?

- Many languages support OOP, e.g. Java and C#.
- But C++ has many subtleties, technicalities and rich OOP features.
- Its implementation and runtime system make it **well-adapted** to the low-level application areas mentioned previously.
- **Prerequisite.** This course aims at teaching C++ to an audience *well-trained in computer programming*.

Sep 2022	Sep 2021	Change	Programming Language	Ratings	Change
1	2	▲	Python	15.74%	+4.07%
2	1	▼	C	13.96%	+2.13%
3	3		Java	11.72%	+0.60%
4	4		C++	9.76%	+2.63%
5	5		C#	4.88%	-0.89%
6	6		Visual Basic	4.39%	-0.22%
7	7		JavaScript	2.82%	+0.27%
8	8		Assembly language	2.49%	+0.07%
12	16		**GO Lang**		
33			**Scala**		

Do not like programming ?

- Tester, quality assurance engineer, …
- System administrator, network engineer, …
- Customer support, configuration, customization, ….
- Database developer, administrator, consultant, …
- Game developer, designer, tester, etc …
- IT governance, auditing, contracting, etc. ….
- Education, training, etc. ….
- …… ***homemaking*** واء واء

Lecture Objectives

1. Introduction
2. Course Objectives
3. Course Administration
4. Programming Paradigms
5. C++ Revision
6. Readings

II. Course Objectives

- Review of C++ Important of Concepts
- Learn basic OOP modeling and programming
- Learn how to implement OOP in C++
- Learn how to organize programs
- Learn advanced programming concepts
- Implement medium size programs
- Work in teams

III. Course Administration

Basic Course Information

- **Course Code: CS213**
- **Course Name: Programming II**
- **Course Credit: 3 credits**
- **Instructor: Dr. Mohammad El-Ramly,**

- **Office Hours: 12:30 – 2:00 Sunday Tuesday**
- **TAs: Many of them – Check your group TA**
- **Site:** http://www.acadox.com/class/64401

Mode of Delivery

- **Lec: Sun & Tues @ 2:30 pm @ Shafee**
  Sun & Tues @ 4:00 pm @ Farag
- **Lab: As scheduled**
- **Recommended Textbook:**
  - **C++ How to Program (10th Ed) (Chap 7 to 18)**
  - Problem Solving with C++ (Walter Savitch)
  - Programming Abstractions in C++ (Eric Roberts)
    http://www-cs-faculty.stanford.edu/~eroberts//CS106B-Reader.pdf
  - www.cplusplus.com
- **Coursework: Assignments, Labs, Exams**

  ⇒ **Expect to put minimum 8 hours a week**

Course Assessment

- Final exam 60 marks
- Midterm ~10 marks
- Quizzes ~8 marks
- Lab tasks ~6 marks
- Assignments ~18 marks

Cheating

- **Do not even think about it**

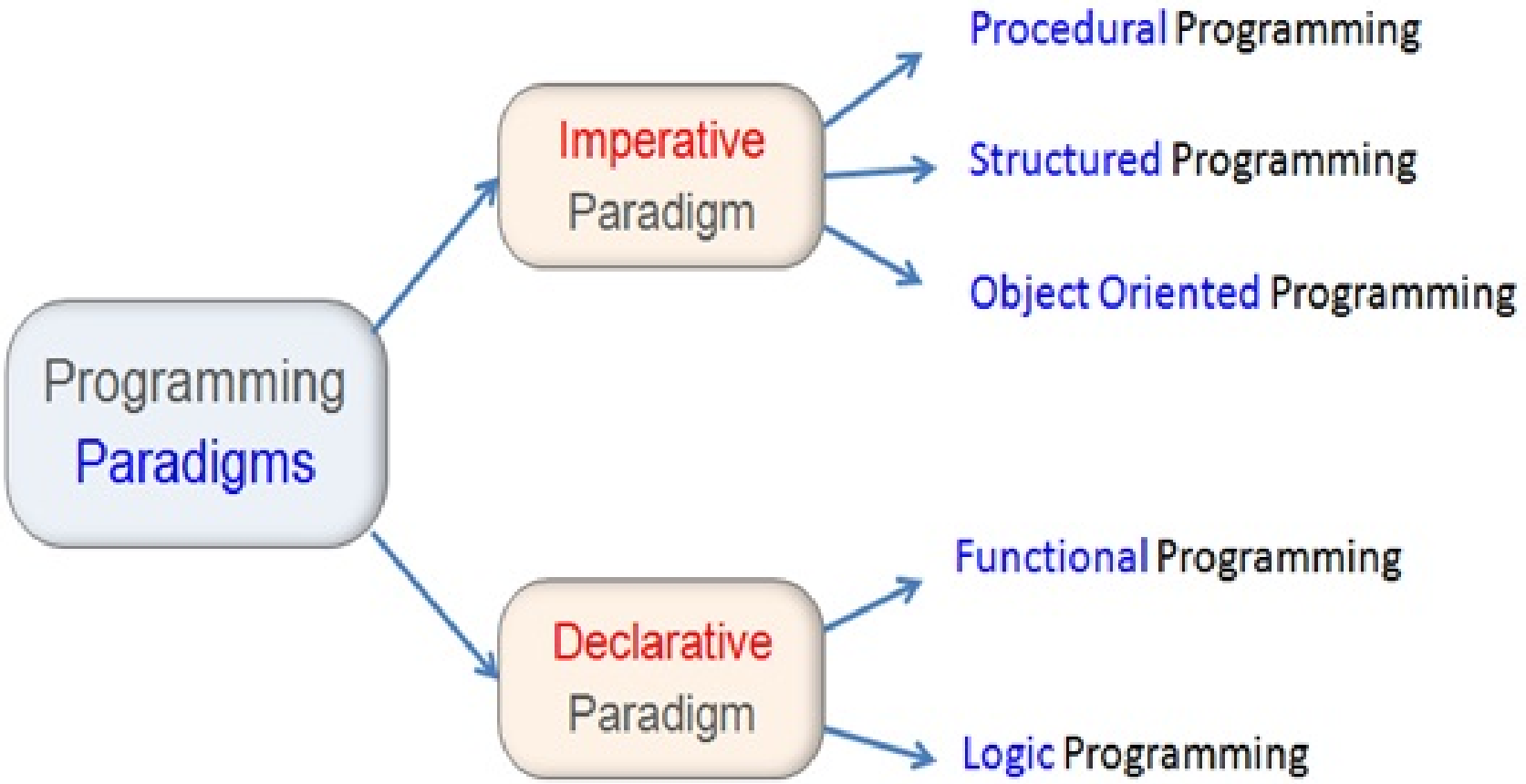
IV. Programming Paradigms

Lecture Objectives

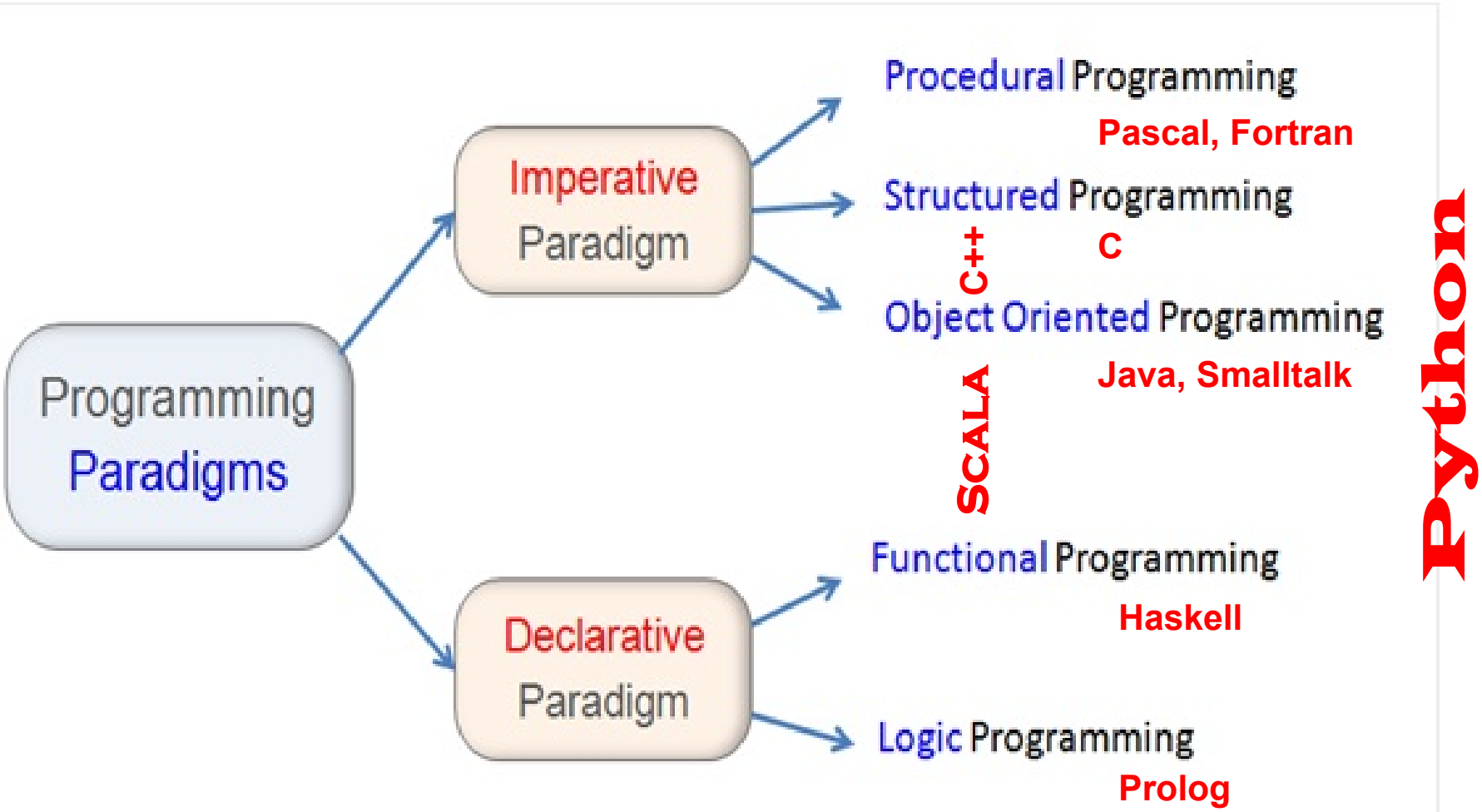
1. Introduction
2. Course Objectives
3. Course Administration
4. Programming Paradigms
5. C++ Revision
6. Readings

- A programming language is a problem solving tool.
- A program is a solution to a problem
- A programming paradigm is a way for organizing the program code.
- A programming language follows one or more programming paradigm

Programming Paradigms



Programming Paradigms



Hello World in Structured C

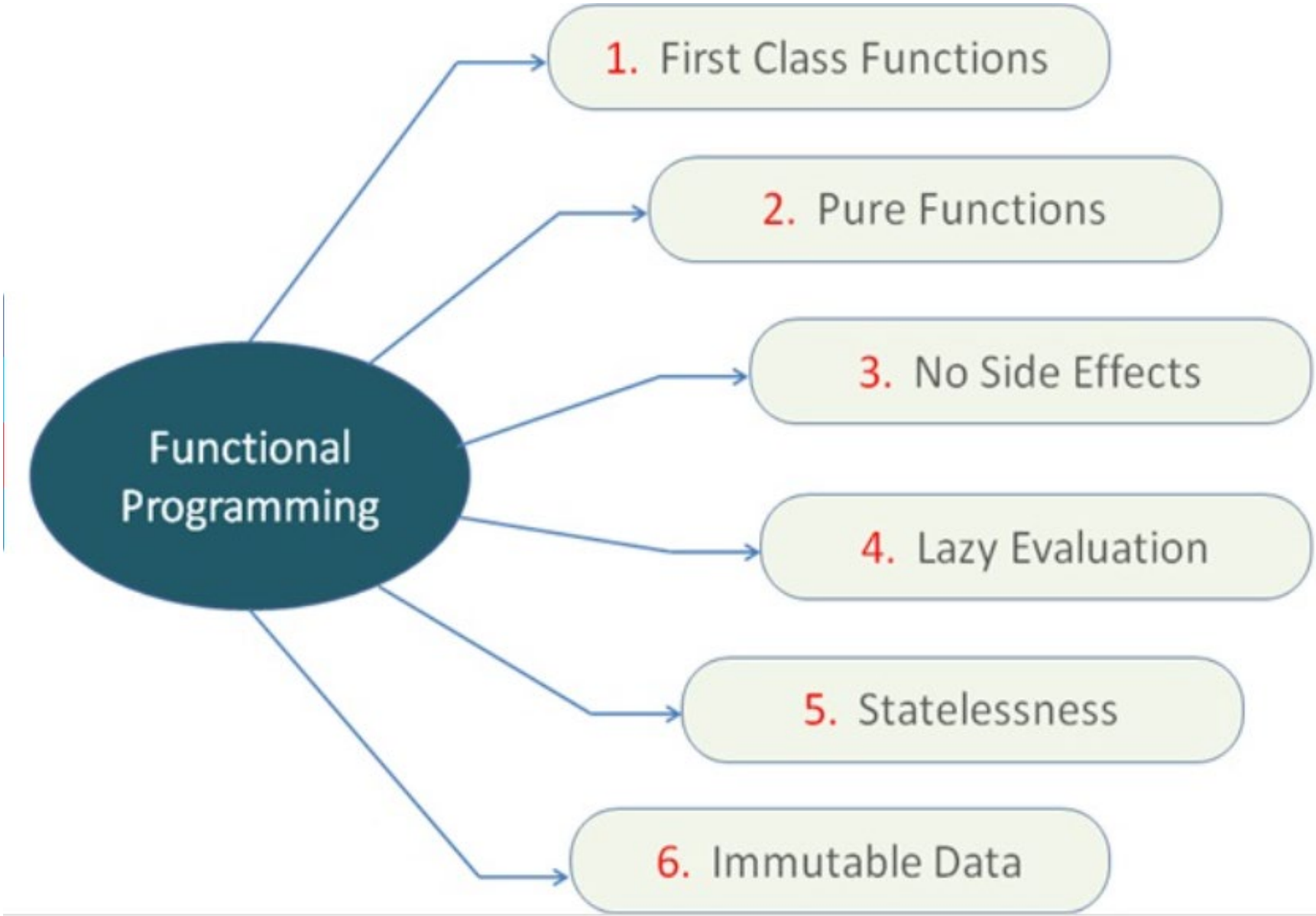
```
#include <stdio.h>

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

Hello World in OOP Java

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

Functional Programming



Procedural Programming VS. Functional Programming

```
int factorial( int n )
{
   int result = 1;

   for ( ; n > 0 ; n--)
   {
      result *= n ;
    }
 return result ;
}
```

```
fac :: Integer -> Integer

fac 0 = 1

fac n | n > 0 = n * fac ( n – 1 )
```

6. Readings

Lecture Objectives

1. Introduction
2. Course Objectives
3. Course Administration
4. Programming Paradigms
5. C++ Revision
6. Readings

Review Programming I Readings

OOP in Scala

```
class Bird
class Cat {
    def catch(b: Bird): Unit = ...
    def eat(): Unit = ...
}

val cat = new Cat
val bird = new Bird
cat.catch(bird)
cat.eat()
```