OOP
Object Oriented Programming

for: SCII/2019 / SCCI 2019 / SCCJ 2019
July – October 2022

By: Salesio M. Kiura
Department of Computer Science and Informatics
School of Computing and Informatics,
Technical University of Kenya (TU-K)

3

Session 3:
OOP Analysis,

14th July 2022

Recap

- Course Objectives, Learning outcomes
- Paradigms and languages, comparison and contrast
- Merits and Demerits of OOP
- Illustrative examples
- Basic OOP concepts
- OOP Analysis: Case studies

EXAMPLE Cases to illustrate OOP Analysis

- 1. TUK innovations
  - Person Identification system (Part of security arrangements initiative)
- 2. Part of a Hospital/ Health Center System
- 3. Part of a Police System

- Automation of a land registry
  - Ref. "Land Registration (General) Regulations 2017" – pages 6 and 7
    - See Part II – of the regulations titled:
      Organization and Administration of Registries
    - ➔ we go through the overview of what happens at the registry
      pages 6 and 7

1

- In the context of the ongoing modernization of TU-K, the University head of Security has approached you to design a simple system for the ***askaris*** manning the University gate. The idea is to automate / computerize what happens at the gate. Starting with the Visitors-Book. The visitors book records all visitors to the University premises. At any given gate, a record of a visitor shows, among other things the following:
  - The officer who attended to the visitor, the date of visit, details of the visitor, destination point/office, the purpose/objective of the visit, mode of traveling used (and corresponding details), report from the visited person, gate used to exit, etc
  - After listening to the visitor, the security officer (***askari***) must indicate on the form whether the purpose of visit is official, private, or returning resident. Official can be administrative office visit, lecturing, studying, working,
- etc

- You are part of a team of developers mandated to come up with a **hospital information management system**. The system will cover the entire processes of attending to patients. From an initial meeting with the head of the medical records at the hospital, you have found out the following.
- … the hospital has a casualty department that acts as the first point of contact with patients. Once a patient arrives at the casualty, his/her details are taken starting with his names. Given the names, the clerk is able to search the records and find out if this is a returning patient or a first time client of the hospital. If the patient is a returning one, he/she pays a registration fee of Kshs. 50 and proceeds to the triage where his/her vital signs (blood pressure, pulse rate, height, age, temperature, etc) are recorded before proceeding to the consultations department to see a clinician who is able to diagnose the ailment of the patient. A first time patient is required to provide more details such as: date of birth, address (where resident), next of kin, etc. he/she pays 100 registration fees and proceeds to the triage from where hr/she is attended to just like a continuing patient. ….

- In the context of the ongoing police reforms in Kenya, the Officer Commanding (a local police) Station has approached you to design a simple system for the station. The idea is to automate/computerize the **police Occurrence Book (OB).** The OB is used to record complaints from the wananchi at the police station. A record of a complaint shows, among other things the following:
  - The police officer who attended to the mwananchi, the date of incidence, details of the person reporting, the report from the mwananchi, etc
- After listening to the mwananchi, the police officer must indicate on the form whether the reported incident is a theft, murder, disagreement, accident or any other

Group Work

- 1. Hospital Information System
- 2. Police OB book
- 3. TU-K

Exercise

- Required:
  - Identify objects
  - identify **relevant** data(members of the class)
  - Identify **relevant** methods (behaviors and messages)

Object Oriented Programming in practice

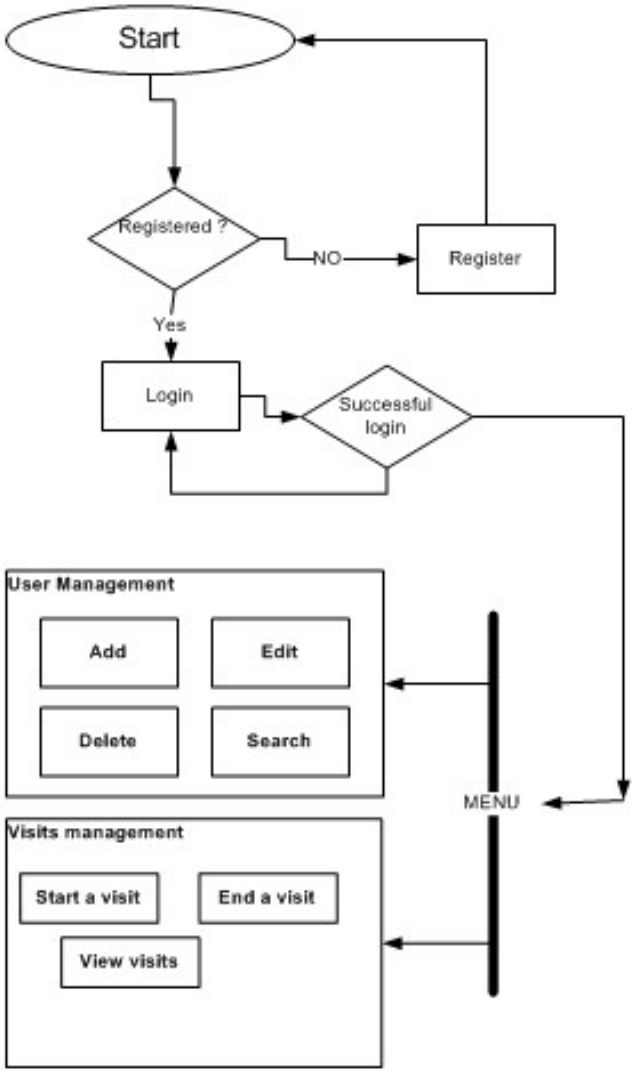
- We said Object Oriented programming is about:
  - objects and assigning responsibilities
  - Objects communicate to other objects by sending messages
  - Messages are received by the methods of an object

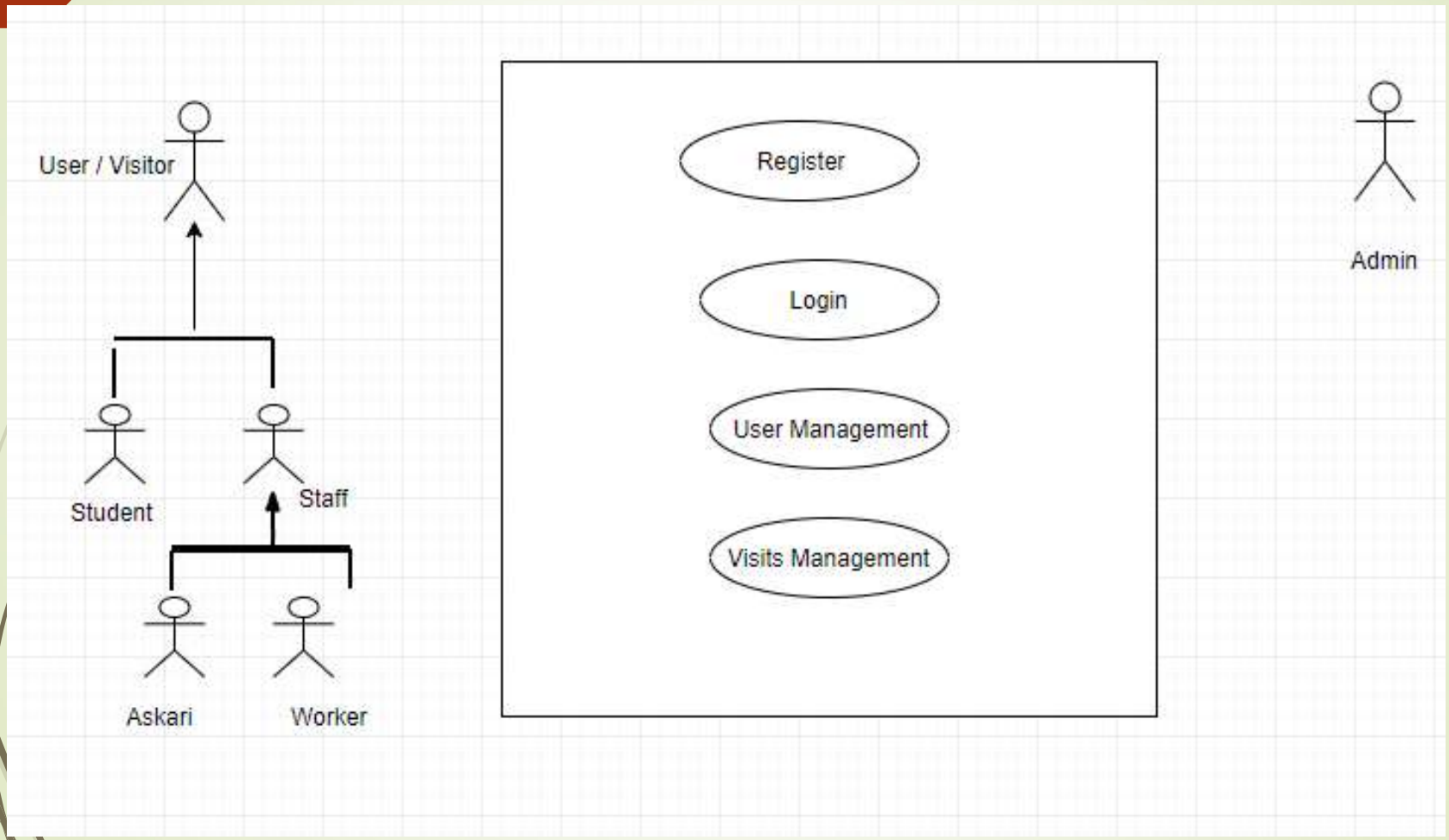
- Your Identified Objects
  - …

Mini projects

- With the above description as an example, you are required to identify and describe a scenario that can form the basis of a semester mini-project
  - Deliverable: A flow chart (don't mind the completeness/correctness at this time)
  - An initial "general overview" class diagram representing the project's case
- Due in next class!
  - Group work

- Example of deliverable




User / Visitor
Student
Staff
Askari
Worker
Register
Login
User Management
Visits Management
Admin

Lab Class Session

Getting started with java (1 of 2) GraalVM

- GraalVM – a high-performance JDK distribution. It is designed to accelerate the execution of applications written in Java and other JVM languages
- Getting GraalVM installed and ready-to-go should take a few minutes
  - https://www.graalvm.org/
  - Downloads (**GraalVM Community** – a Community supported **open source** build)

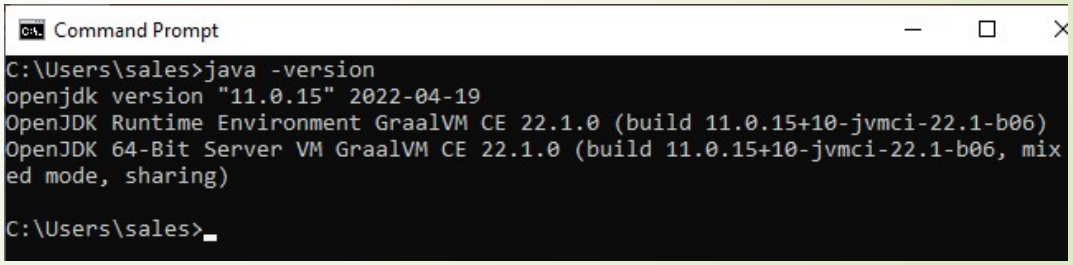
Download Link

- Direct download Link:
  - https://github.com/graalvm/graalvm-ce-builds/releases/tag/vm-22.1.0
- This distribution of GraalVM Community includes:
  - Java runtime with the GraalVM compiler
  - JavaScript runtime
  - LLVM runtime
  - Monitoring and debugging tools
- The GraalVM environment can be extended with optionally available components for web development environments and frameworks

Platform	Java 11	Java 17	
Linux (amd64)	download	download	*instructions*
Linux (aarch64)	download	download	*instructions*
macOS (amd64) †	download	download	*instructions*
macOS (aarch64/M1) †	download	download	*instructions*
Windows (amd64)	download	download	*instructions*

Installation guides

- Windows:
  - Installation on Windows Platforms (graalvm.org)
    - https://www.graalvm.org/22.1/docs/getting-started/windows/
- Linux platform
  - Installation on Linux Platforms (graalvm.org)
    - https://www.graalvm.org/22.1/docs/getting-started/linux/
- Others:
  - https://www.graalvm.org/22.1/docs/getting-started/
- Test the installation of JDK
  - java - version



Text based HelloWorld App execution

- In VS code
  - With a text editor:

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

  - C:\your_Folder> **javac** HelloWorld.java
  - C:\your_Folder> **java** HelloWorld

    Hello World Class!

Getting started with java NetBeans

- Apache NetBeans (https://netbeans.apache.org/)
  - Fits the Pieces Together,
  - **Development Environment**, Tooling Platform and Application Framework.
    - More than an editor!
  - Download links (choose a version of your choice)
    - https://netbeans.apache.org/download/
  - Installation is a standard process
    - The system should recognize the earlier jdk folder (e.g. GraalVM)

Getting started with java (1 of 2)

- Objectives:
  - Be able to create projects in Netbeans
  - Write the first java program: HelloWorld Application

Getting started with java (2 of 2)

- Environment
  - Netbeans
  - Java
- Follow the instructions as guided in class to create a new Class

- Downloading (https://netbeans.org/downloads/xx.xx/) Version
- Installing
- Running the IDE
  - General Overview of Netbeans
  - Hello World Application
  - Control Structures***

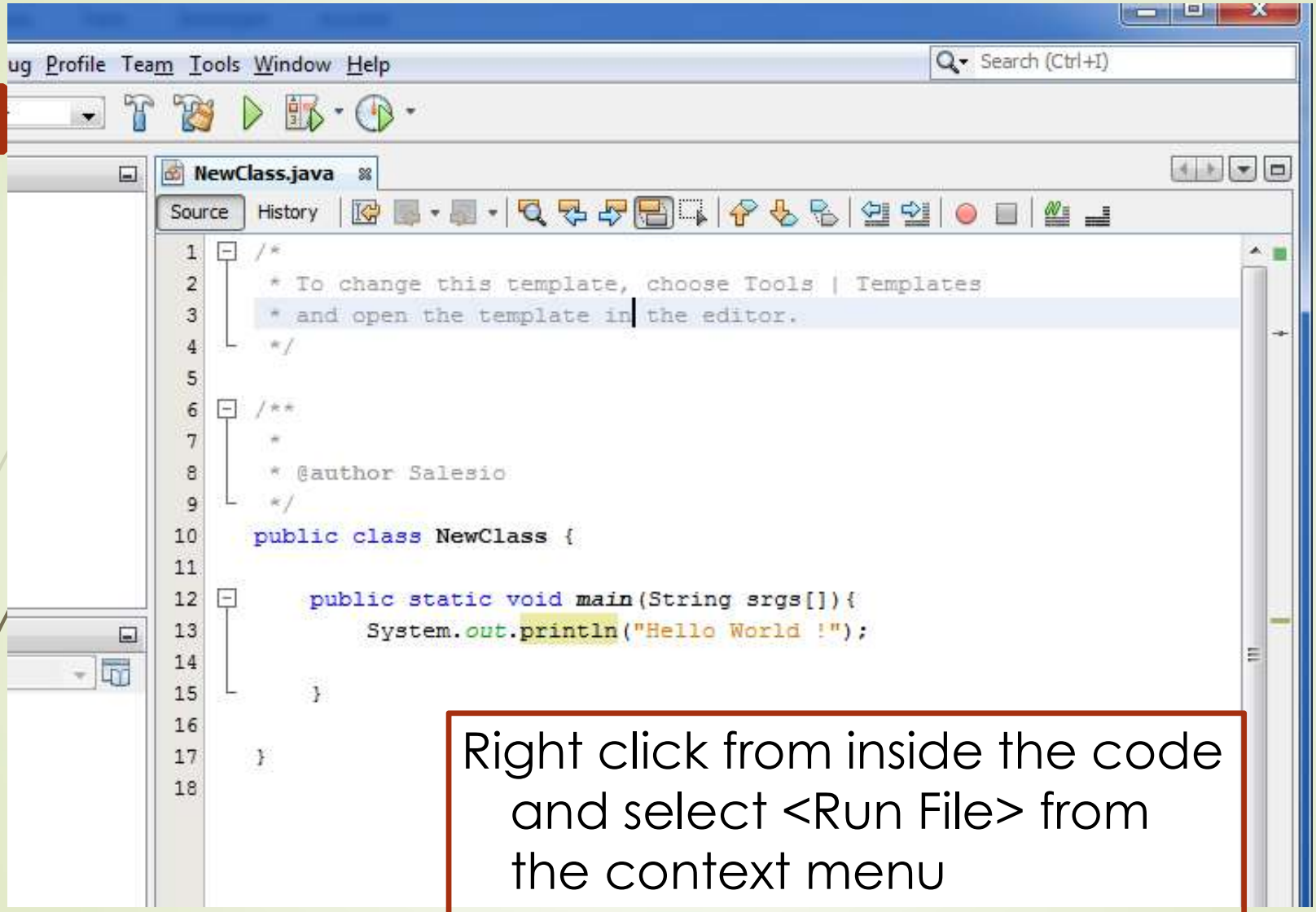
HelloWorld Code

```
public class HelloWorld {

    public static void main(String srgs[]){
        System.out.println("Hello World !");

    }

}
```



Right click from inside the code and select <Run File> from the context menu

- Thanks
  - Please share your feedback and comments

About JAVA

- Is used for creating:
  - intelligent consumer-electronic devices (cell phones)
  - Web pages with dynamic content
  - large-scale enterprise applications

Java life cycle

- Java programs normally undergo four phases:
  - **Edit (**Source code (.java)**)**

    Programmer writes program (and stores program on disk)
  - **Compile (**Byte codes (.class) , as (.exe) in c++

    Compiler creates bytecodes from program (.class as .exe in c^{++})
  - **Load**

    Class loader stores bytecodes in memory
  - **Execute**

    Interpreter: translates bytecodes into machine language

- **Other concepts**
  - The Java Application Programming Interface (API)
    - a large collection of ready-made software components. It is grouped into libraries of related classes and interfaces; these libraries are known as packages.
    - E.g. System.out.*; java.util.*
  - Java Virtual Machine (JVM)
  - Machine code (platform dependent)

Simple Exercise

- All participants to "AT LEAST" be able to code:
- Hello WORLD in JAVA!
  - Using Netbeans
  - Hand coded Hello World
    - On command prompt
    - On a GUI (message Dialog Box)
  - Interactive Hello World
    - Enter on command prompt
    - Enter using an input dialog box

Next ➔

Anatomy of a Java program, Objects, constructors