



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)

2



Session 2:

Basic OOP Concepts & Getting Started with oop Analysis

7th July 2022



What is Object Oriented Programming?



■ About:

- objects and assigning responsibilities
- Objects communicate to other objects by sending messages
- Messages are received by the methods of an object



■ What are objects?

- an object represents an individual, identifiable item, unit, or entity, either real or abstract, with a well-defined role in the problem domain.
- An "object" is anything to which a concept applies

OOP is a Paradigm

- Object-Oriented Style
 - Involves programming with *Abstract Data Types*
 - ADTs specify/describe behaviors.
 - The basic program unit is a **Class**
 - Implementation of an ADT.
 - Abstraction is enforced by encapsulation.
 - The basic Run-time unit is an **Object**
 - Instance of a class.
 - Has an associated *state*.

Identifying Objects in a problem Domain

- Objects
 - An object is like a black box. The internal details are hidden.
 - Tangible Things as a car, printer, ...
 - Roles as employee, boss, ...
 - Incidents as flight, overflow, ...
 - Interactions as contract, sale, ...
 - Specifications as colour, shape, ...



Benefits of an Object Oriented Approach

- Why do we care about objects?
 - **Modularity** - large software projects can be split up in smaller pieces.
 - **Re-usability** - Programs can be assembled from pre-written software components.
 - **Extensibility** - New software components can be written or developed from existing ones.

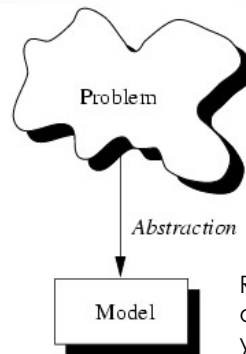


Basic terminologies/concepts of OOP

- Abstraction
- Encapsulation
- Inheritance
- Polymorphism
- Aggregation
- **Behaviour and Messages !**

Basic terminologies/concepts of OOP

- **Abstraction** is the representation of the essential features of an object. These are 'encapsulated' into an *abstract data type*.



Create a model from a problem with abstraction.

Represents **(relevant) features** of the problem domain (from your perspective).

How Abstraction takes effect in Programming

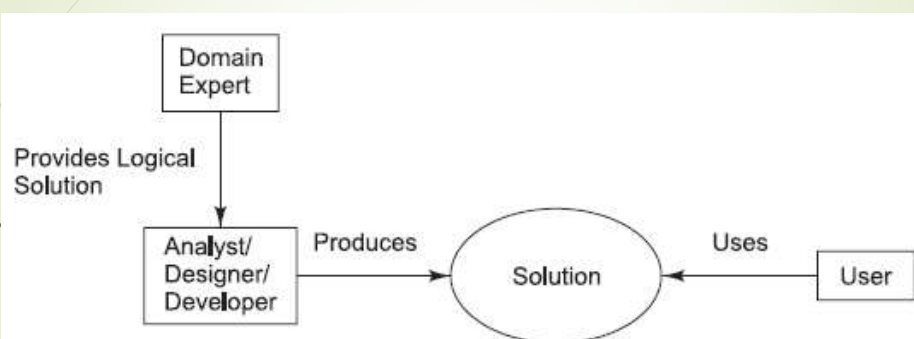



Fig. 1.3 People associated with the solution



Encapsulation

- **Encapsulation** is the practice of including in an object everything it needs hidden from other objects. The internal state is usually not accessible by other objects
 - Black box view
 - Analogy to a human given an instruction, A Car ignition, etc
 - Reality in modern systems: Send money via phone, send email, etc



Inheritance

- **Inheritance** means that one class acquires the characteristics of another class.
 - This is also called a “is a” relationship:
 - A car *is a* vehicle
 - A dog *is an* animal
 - **A student is a person | A lecturer is a person → what does this mean?**
 - Etc.

Polymorphism

- **Polymorphism** means “having many forms”. It allows different objects to respond to the same message in different ways, the response specific to the type [instance] of the object.
- E.g.
 - Imagine a scenario in a university system whereby a Person Object can take the forms of either a student or a lecturer. A call to the instance of the Person object, requiring an identification number relevant to the university, will yield different values/types as follows
 - In one case a number of the format SCxx/0xxxx/20xx
[Person has taken Student form]
 - In another case a number of the format AC0xxxx []
[Person has taken Lecturer/Academic staff form]
 - NB. There is a relationship between Inheritance and Polymorphism!

Aggregation

- **Aggregation** describes a “has a” relationship. One object is a part of another object.
- E.g.
 - A Bus/car has wheels
 - A company has departments
- We distinguish between:
 - *composite* aggregation (the composite “owns” the part)
 - *shared* aggregation (the part is shared by more than one composite).

Types of Aggregation

- **Basic** aggregation
 - In the relationship, the child class instance can outlive its parent class
- **Composition** aggregation
 - a child class's instance lifecycle is dependent on the parent class's instance lifecycle
 - a parent class instance will always have at least one child class instance. when the parent instance is removed / destroyed, the child instance is automatically removed/destroyed.
 - A part (child) class instance can only be related to one instance of the parent class

EXAMPLES?

i) A Page has headers and footer sections.
ii) A university faculty has departments.
→ Which aggregation is which?

Behaviour and Messages

- **Behaviour and Messages**
 - The most important aspect of an object is its *behaviour* (the things it can do).
 - Behaviours are implemented as methods/"functions" of a class.
 - A behaviour is initiated by sending a *message* to the object (i.e calling / invoking a method)



Basic terminologies/concepts of OOP



■ Basic terminologies/concepts of OOP

1. Abstraction
2. Encapsulation
3. Inheritance
4. Polymorphism
5. Aggregation
6. **Behaviour and Messages !**



■ Questions?

- ➔ In our Laboratory sessions, we should try and see these functionalities in action! (Using Java)

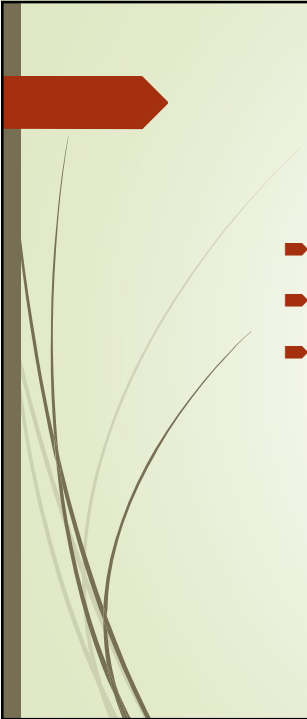
EXAMPLE

- Part of a Police System ?
- Part of a Health Center System
- TUK innovations
 - Person Identification system (Part of security arrangements initiative)
- 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

- 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

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

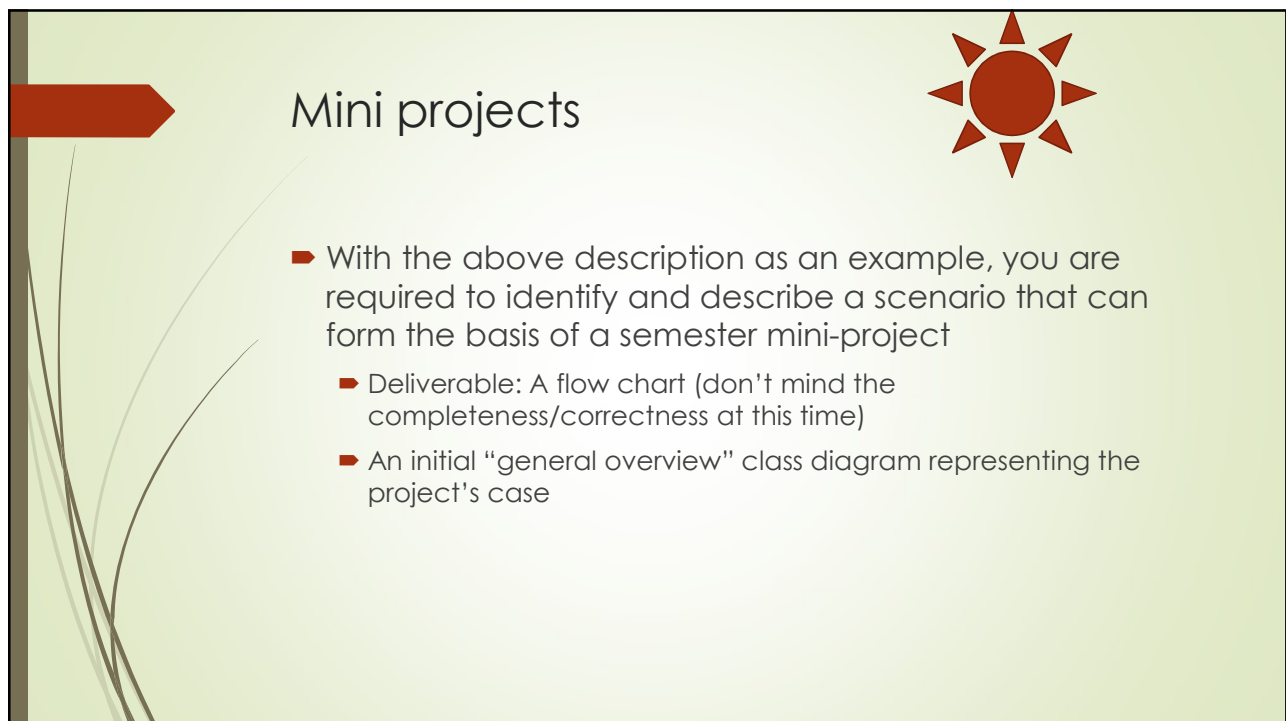
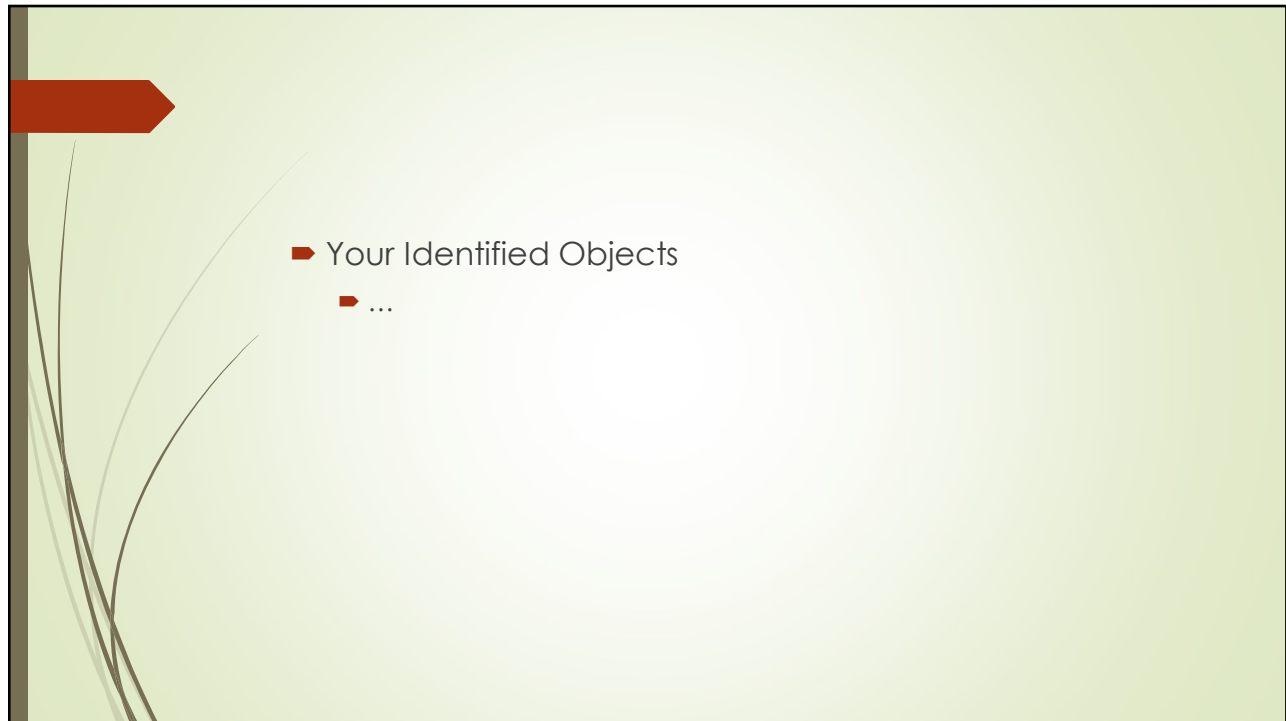


Exercise


- 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

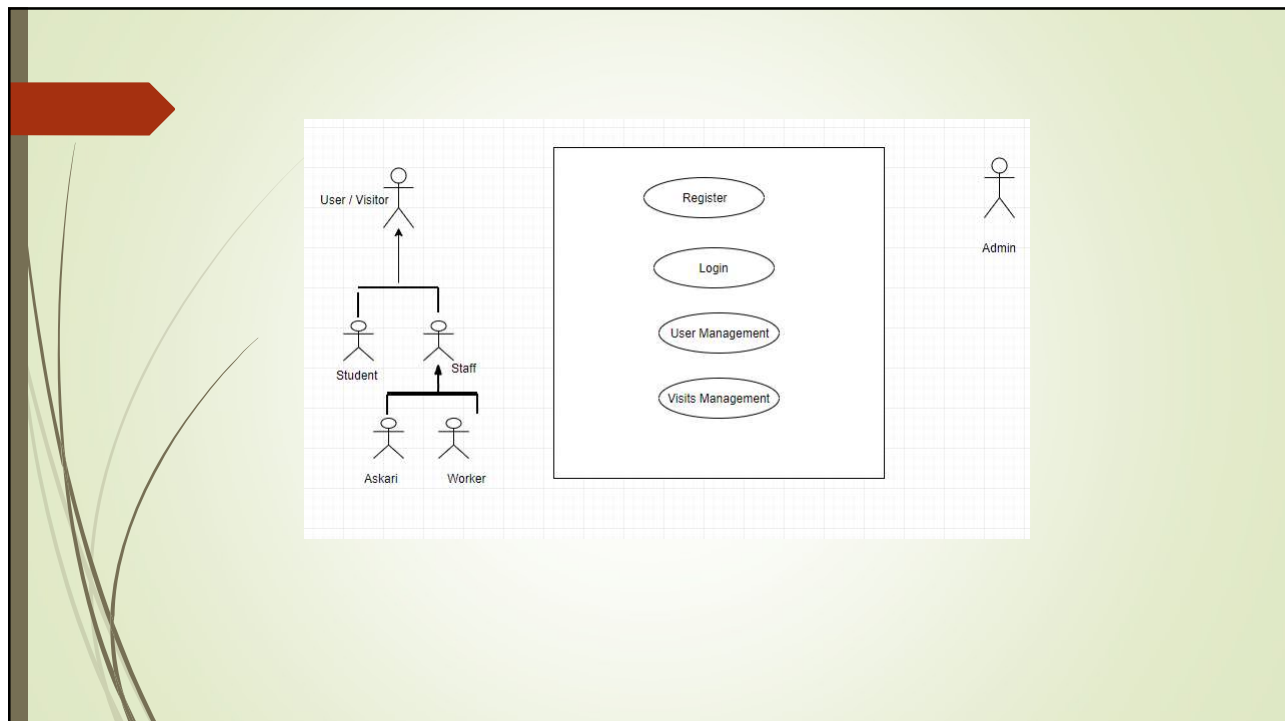
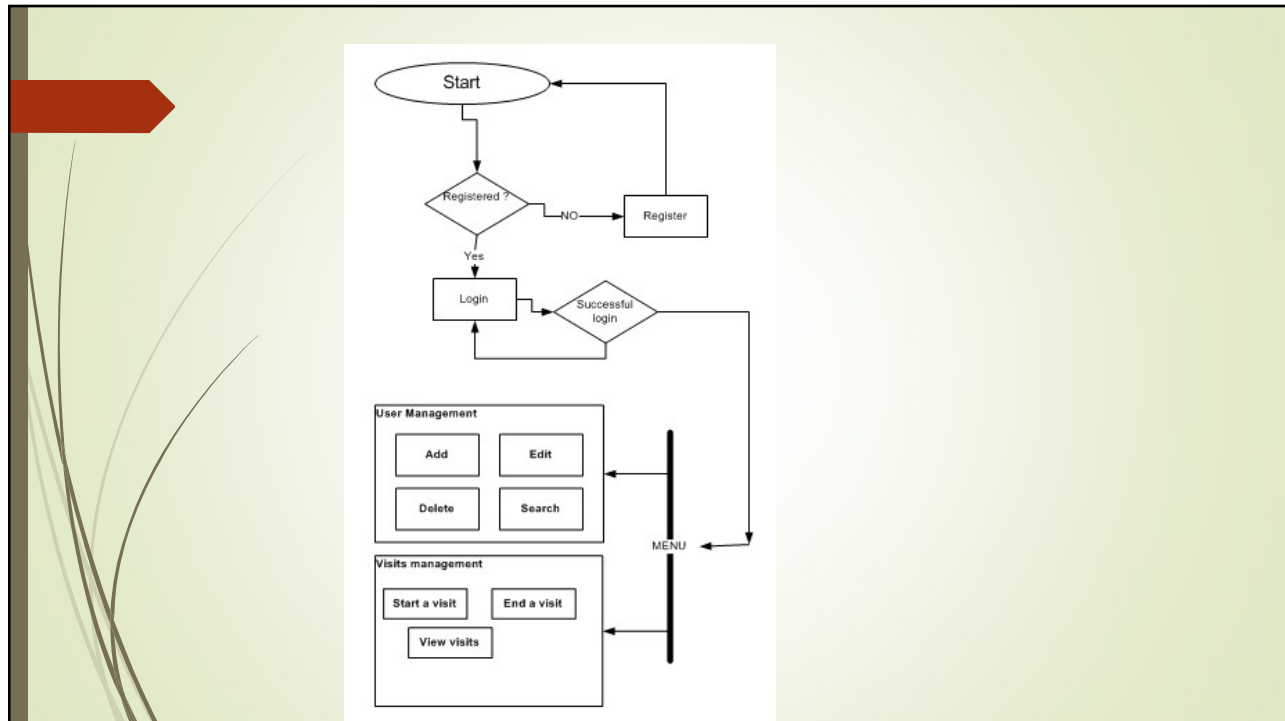


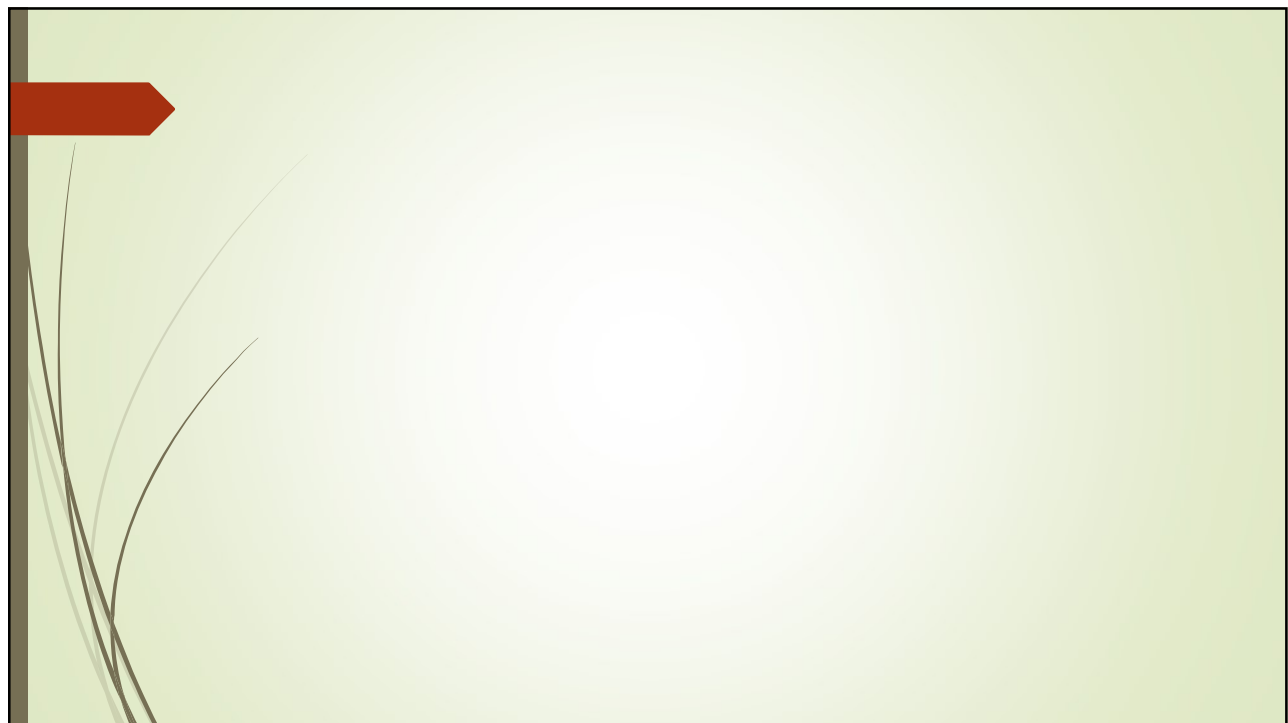
Mini projects



- 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

This slide features a light green background with a dark green vertical bar on the left. A red arrow points right from the bar. Faint, thin lines resembling grass or reeds are visible on the left side. The title 'Mini projects' is centered. To the right of the title is a red sun icon with a circular center and eight triangular rays. Below the title, a red square bullet point is followed by a paragraph. This paragraph is followed by two indented red square bullet points.







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)

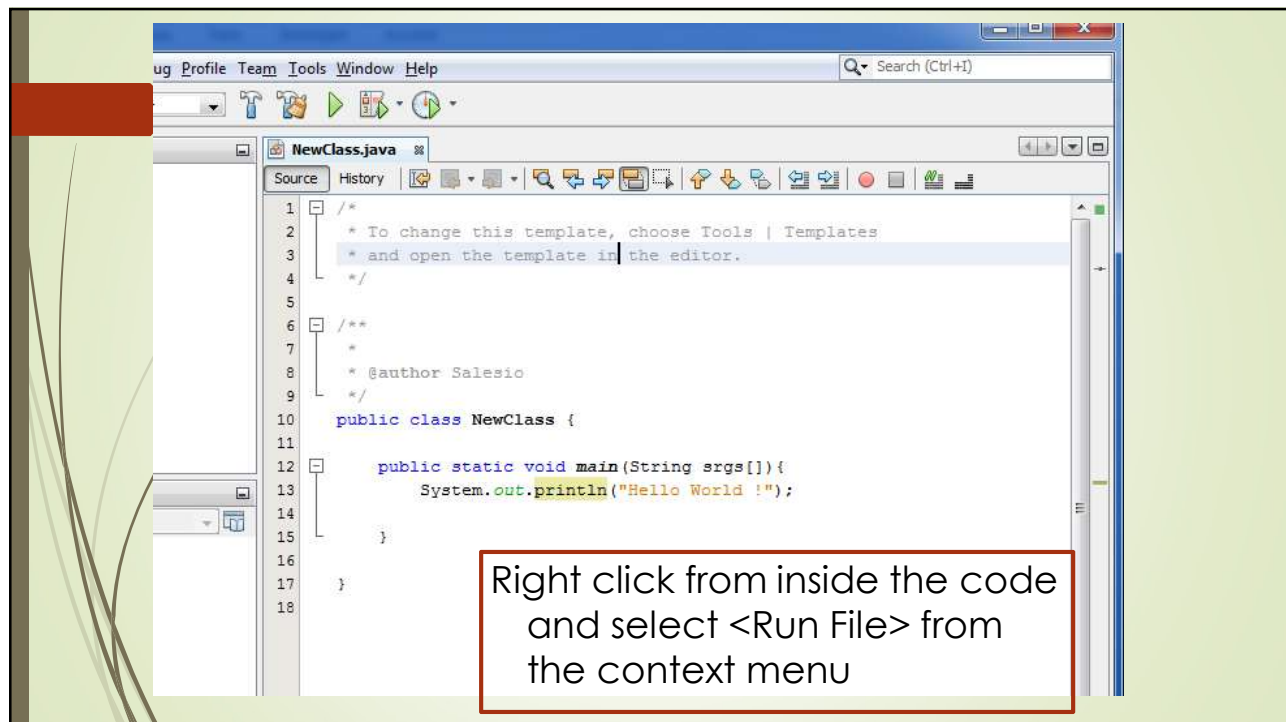
- Environment
 - Netbeans
 - Java
- Follow the instructions as guided in class to create a new Class

- 
- Downloading (<https://netbeans.org/downloads/xx.xx/>)
 - 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 !");  
  
    }  
  
}
```



- 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

