



Object Oriented Programming

ICT2122

Introduction to Object Oriented Programming

P.H.P. Nuwan Laksiri
Department of ICT
Faculty of Technology
University of Ruhuna

Lesson 01

Course Outline

- **Primitive vs non primitive data types**
- **Operators, statements, control structures**
- **APIs**
- **The influence of basic OO principles, including abstraction, encapsulation, inheritance, and re-use, on design and implementation of OO programs.**
- **Essentials of Class Diagrams and classes:**
 - classes, members, methods, constructors
- **Inheritance, abstraction, polymorphism encapsulation**
 - subclasses, this, super, Object etc
 - concrete, abstract
 - static and dynamic polymorphism
 - getters and setters
- **Abstract class and Interfaces**

Course Outline

- **Access control**
 - Public, private, protected, default
- **String handling**
- **Exception handling:**
 - exceptions, throw, try/catch, throws
- **Interactions with DBMSs**
- **Introduction to GUI with AWT/Swing**
- **Files and Streams handling:**
 - text input (Scanner / Reader), binary input (DataInputStream, ObjectInputStream, serialization), output: (PrintStream, DataOutputStream, ObjectOutputStream, PrintWriter).
- **Introduction to Design Patterns**
 - Singleton
- **Introduction to Threads**
 - Thread class, Runnable interface

Course Plan

- Lectures – 02 Hours per Week
- 02 Credits
- Theory
- Evaluation
 - Quizzes
 - Mid Term Evaluation

Eligibility and Evaluation Criteria

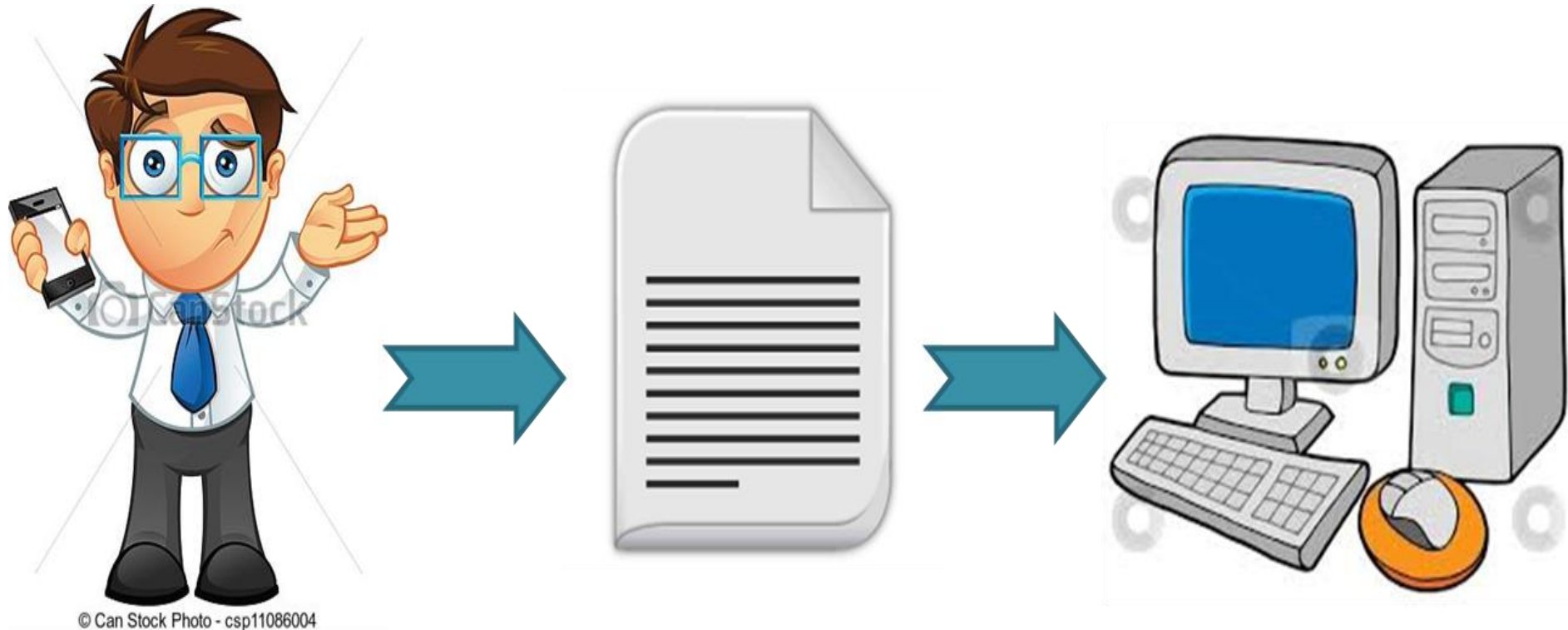
- Eligibility
 - **80%** Attendance is **MANDATORY**
 - 50% from CA marks is **MANDATORY**
- Evaluation Criteria
 - CA 30%
 - 10% From Quizzes
 - 20% Mid Term Theory Evaluation
 - ESA 70%
 - 70% From Final Exam
(Theory Paper)

Outline

- What is Object Oriented Programming
- Fundamentals of Object Orientation
- Why Object Orientation ?
 - Modularity
 - Information-hiding
 - Code re-use
 - Pluggability and debugging ease
- Understanding Classes and Objects
- Real-World Scenario
- Class
- Object
- Instance
- Instantiation

What is Programming?

- The process of writing computer programs



Programmer

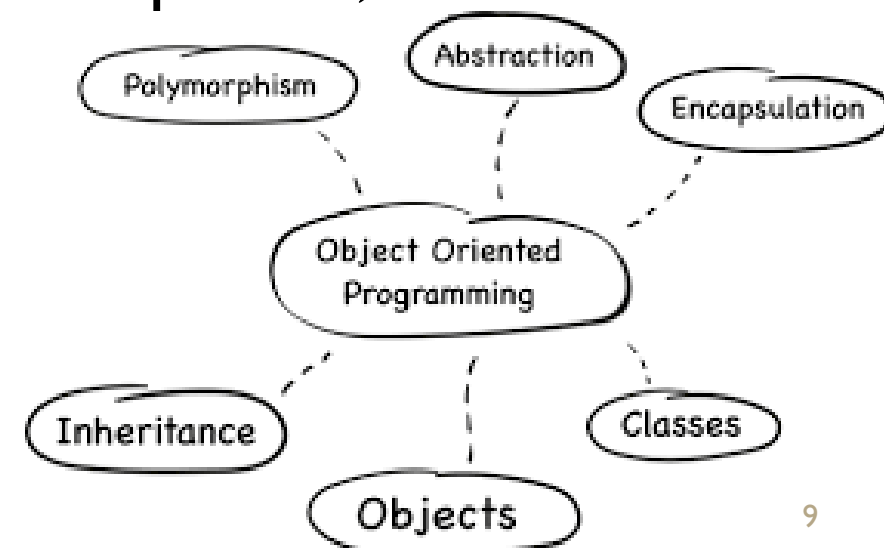
Program Code

Programming vs. Development vs. Engineering

- Programming?
- Development?
- Engineering?

What is Object Oriented Programming

- Object-oriented programming
 - is a paradigm
 - based on the concept of wrapping pieces of data, and behavior related to that data,
 - into special bundles called objects,
 - which are constructed from a set of “blueprints”,
 - defined by a programmer,
 - called classes.



Fundamentals of Object Orientation

A program is viewed as

- A collection of objects
 - Objects pass messages to each other
 - Each object decided what to do with a received message
- It is more meaningful to talk about an object-oriented system than a program
 - An object-oriented system is a set of interacting objects organized into classes

Fundamentals of Object Orientation

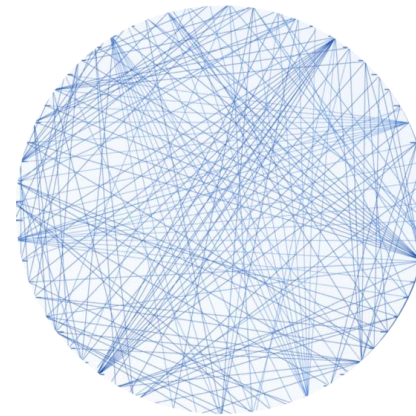
- Focus is on data and not on function
 - Definition of data and its attribute and how it will be manipulated is the focus
 - Exact mechanism of manipulation (procedure or algorithm) is not a primary focus

Why Object Orientation ?

- For keeping large software projects manageable by human programmers
 - Modularity
 - Information-hiding
 - Code re-use
 - Pluggability and debugging ease

Modularity

- The source code for an object can be written and maintained independently of the source code for other objects.
- Once created, an object can be easily passed around inside the system.



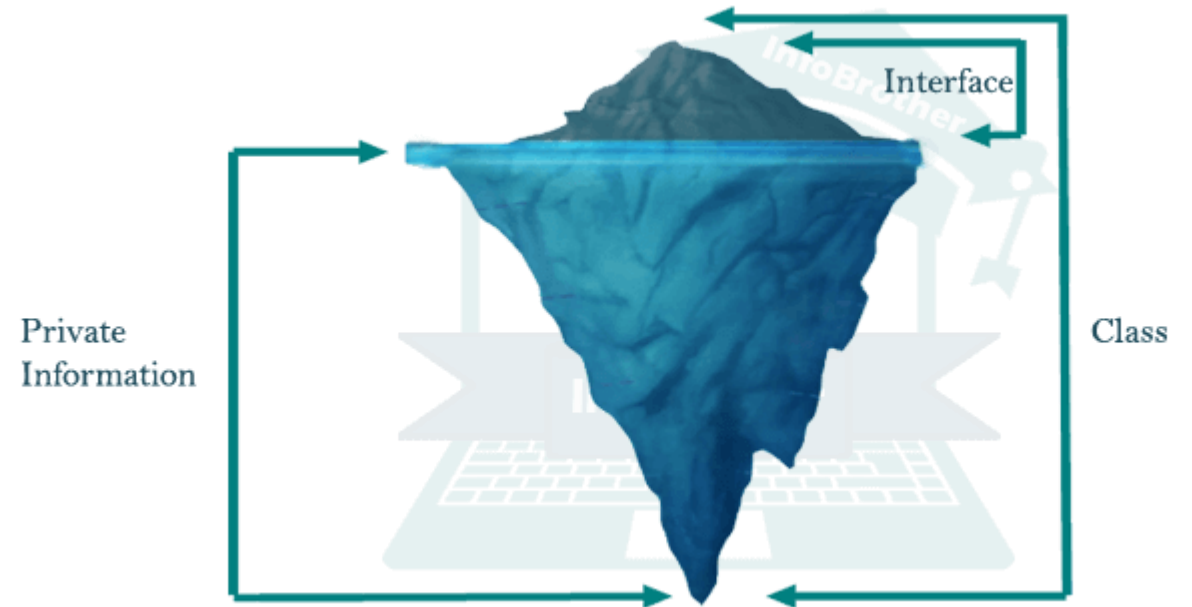
Non-modular



Modular

Information-hiding

- By interacting only with an object's methods, the details of its internal implementation remain hidden from the outside world.



Code re-use

- If an object already exists (perhaps written by another software developer), you can use that object in your program.
- This allows specialists to implement/test/debug complex, task-specific objects, which you can then trust to run in your own code.



Pluggability and debugging ease

- If a particular object turns out to be problematic, you can simply remove it from your application and plug in a different object as its replacement.
- This is analogous to fixing mechanical problems in the real world. If a bolt breaks, you replace it, not the entire machine.



Understanding Classes and Objects

A class is like a cookie cutter; it defines the shape of objects

Objects are like cookies; they are **instances** of the class



Photograph courtesy of [Guillaume Brialon](#) on Flickr.

Now you know...

- Class
- Object
- Instance
- Instantiation
- Let's try to go for proper understanding and definitions in the latter part of this lecture.

A Real-World Scenario

“ ...customers are allowed to have different types of bank accounts, deposit money, withdraw money and transfer money between accounts.”

- Start with, “Procedural” approach....

A Real-World Scenario

- Procedural Approach

```
bool MakeDeposit(int accountNum, float amount);  
float Withdraw(int accountNum, float amount);
```

```
struct Account {  
    char *name;  
    int accountNum;  
    float balance;  
    char accountType;  
};
```

A Real-World Scenario

- Procedural Approach
 - Focus is on procedures
 - All data is shared: no protection
 - More difficult to modify
 - Hard to manage complexity

A Real-World Scenario

“ ...customers are allowed to have different types of bank accounts, deposit money, withdraw money and transfer money between accounts.”

- Now, “Object Oriented” approach....

A Real-World Scenario

- Object Oriented Approach

" ... **customers** are allowed to have different types of **bank accounts**, deposit **money**, withdraw **money** and transfer **money** between **accounts**. "

Nouns represent objects
(classes) in the domain.

A Real-World Scenario

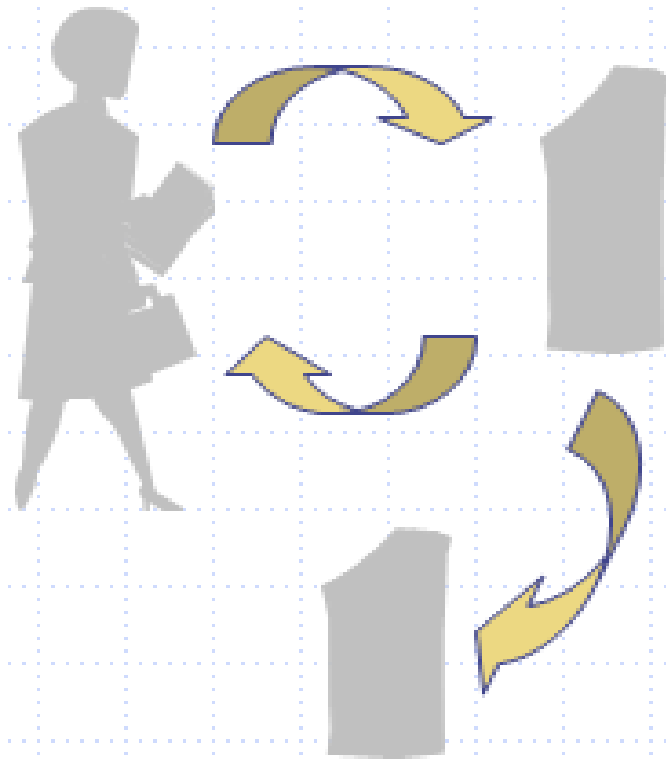
- Object Oriented Approach

" ... customers are allowed to have different types of bank accounts, deposit money, withdraw money and transfer money between accounts. "

Verbs represent actions
(methods) on the objects

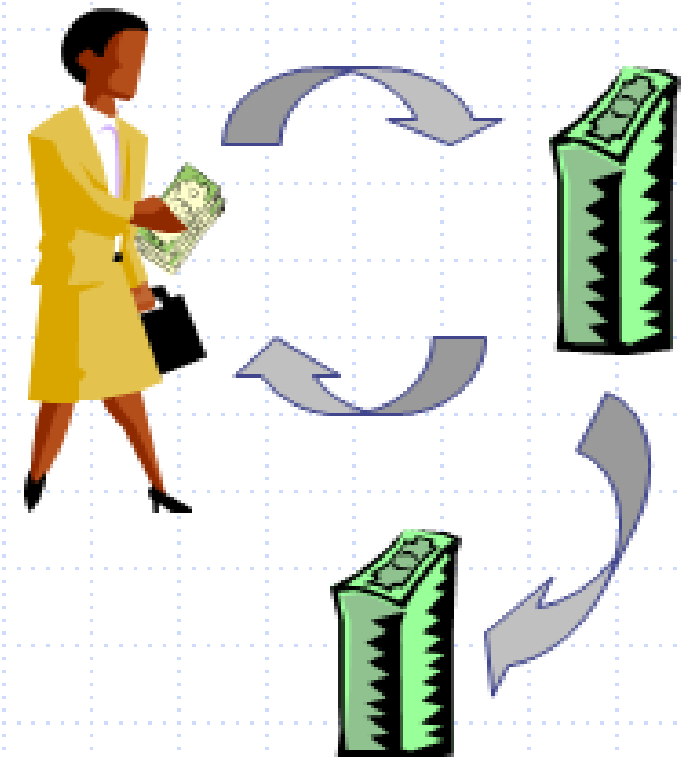
Procedural Vs Object Oriented

Procedural



Withdraw, deposit, transfer

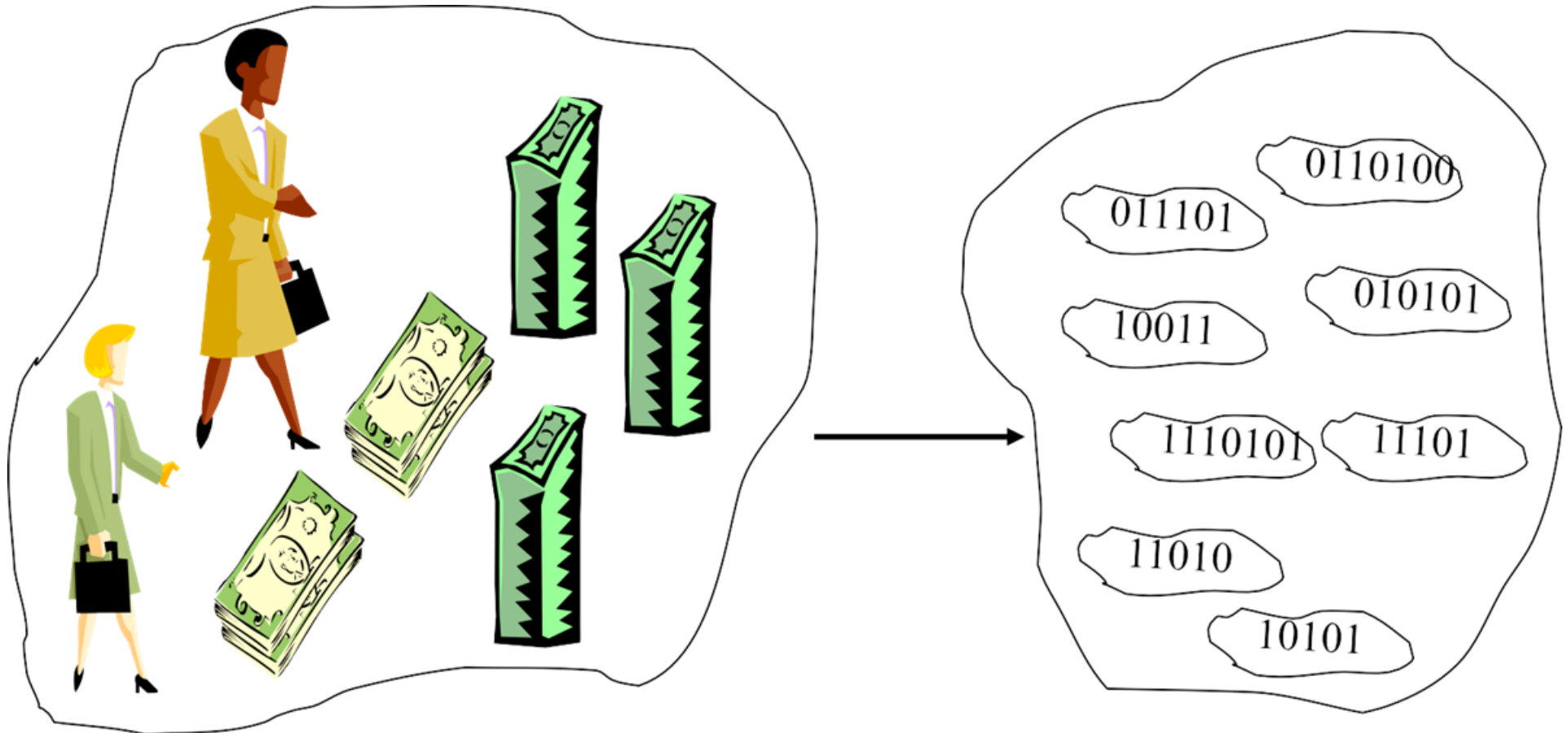
Object Oriented



Customer, money, account

Mapping the world to software

- Objects in the problem domain are mapped to objects in software



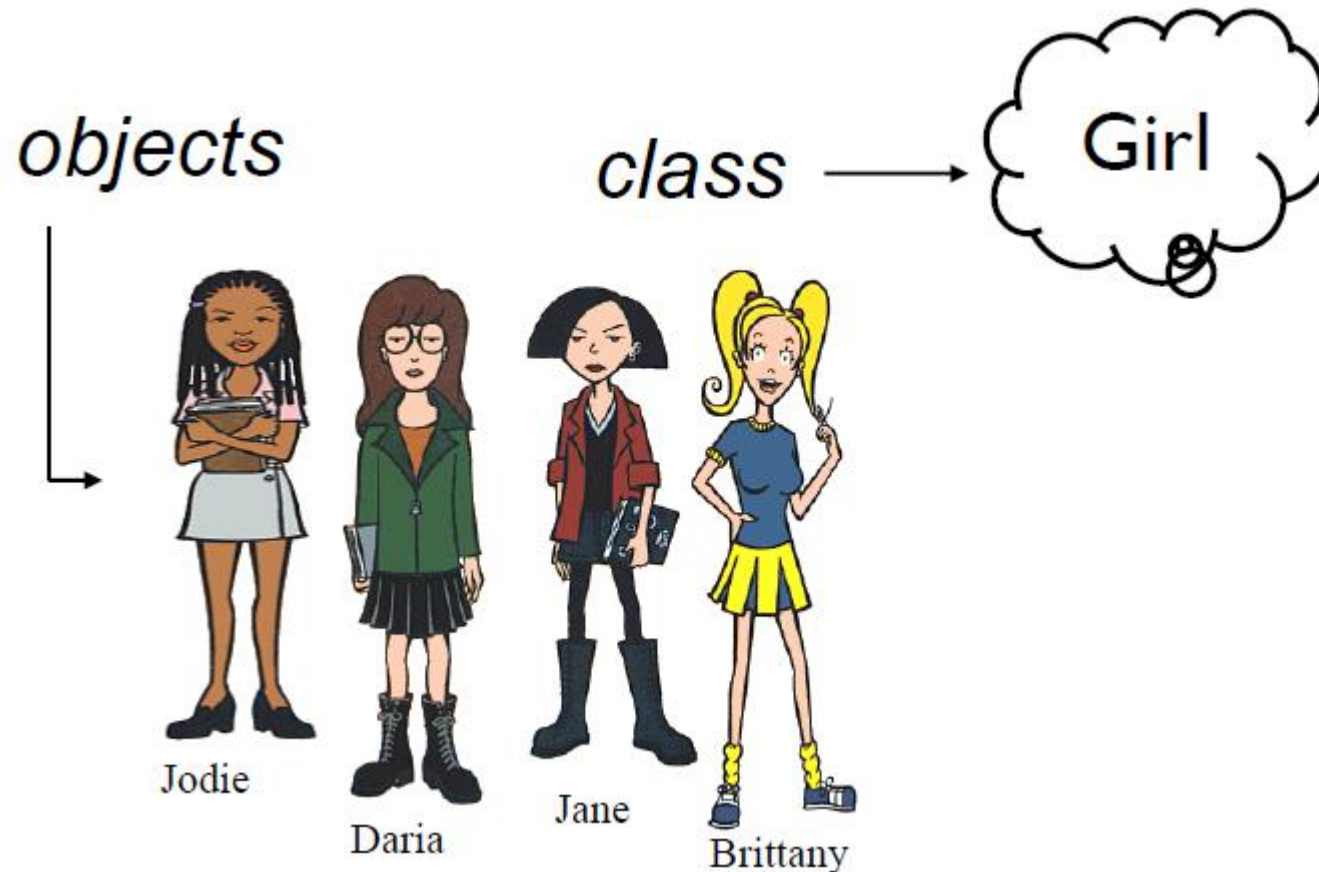
Object Oriented Approach

- Data and operations are grouped together.



Objects and Classes

- Classes reflect concepts, objects reflect instances that embody those concepts.



Objects as instances of Classes

- The world conceptually consists of objects
- Many objects can be said to be of the same type or class
 - My bank account, your bank account, Bill Gates' bank account
- We call the object type a class
 - Type of my bank account is BankAccount
 - Type of bill gate's bank acc is BankAccount

Instantiation

- An Object is instantiated from a Class

```
BankAccount myAccount;  
myAccount = new BankAccount();
```

```
BankAccount gatesAccount;  
gatesAccount = new BankAccount();
```

Objects and Classes

- Class
 - Visible in source code
 - The code is not duplicated
- Object
 - Own copy of data
 - Active in running program
 - Occupies memory
 - Has the set of operations given in the class

What Is an Object?

- Object is an instance of a class.

<https://docs.oracle.com/javase/tutorial/java/concepts/object.html>

Characteristics of Objects

- State (the properties n their values)
 - Dog : name, color, breed, etc...
 - Bank Account: balance, interest rate, etc...
- Behavior
 - Defines interaction with the outside world. (Methods)
 - Dog: making sound (barking), wagging tail, etc...
 - Bank Account: Withdraw, deposit, etc..
- Identity
 - How to differentiate two objects of the same class.
 - Eg: ID, Account Number, Serial No, etc...

How to start...

- Think the real world, be natural.
- Identify objects
- For each object
 - Think what possible states can this object be in?
 - Think what possible behavior can this object perform?

Starting objects...

- Alan Kay's 5 rules
 - Everything is an object
 - A program is a bunch of objects telling each other what to do by sending messages
 - Each object has its own memory made up of other objects
 - Every object has a type
 - All objects of a particular type can receive the same messages (invoke methods)

What Is a Class

- A blueprint of an object.

<https://docs.oracle.com/javase/tutorial/java/concepts/class.html>

Class

- In the real world, you'll often find many individual objects example, the same kind.
- For example, take bicycles
 - There may be thousands of other bicycles in existence, all of the same make and model.
 - Each bicycle was built from the same set of blueprints and therefore contains the same components.
 - In object-oriented terms, we say that your bicycle is an instance of the class of objects known as bicycles.
 - A class is the blueprint from which individual objects are created.

Instance

- An instance is a specific realization of any object.
- An object may be different in several ways, and each realized variation of that object is an instance.

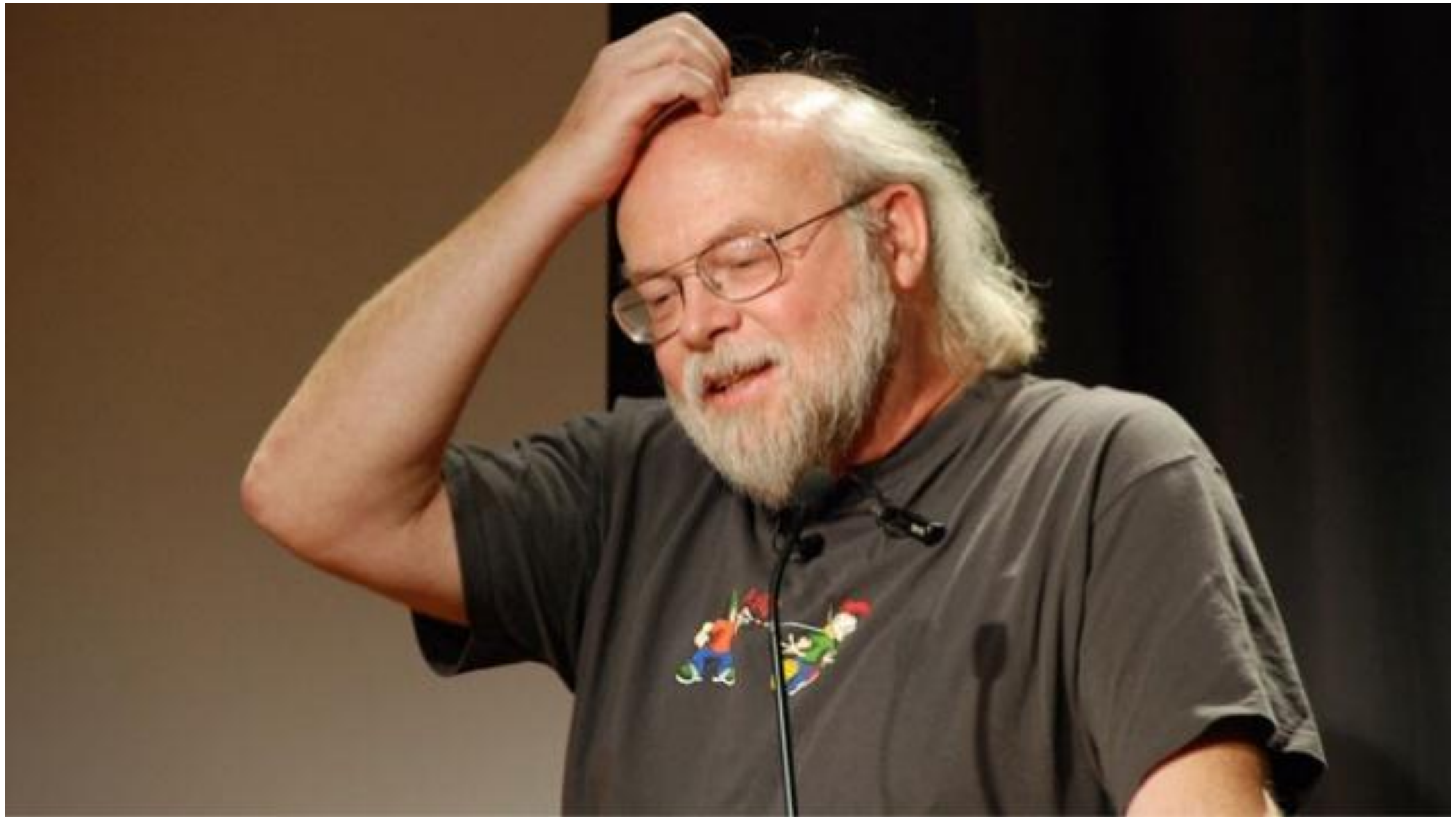
Instantiation

- Instantiation is the creation of a real instance or particular realization of an abstraction or template, such as a class of objects or a computer process.

(The creation of a realized instance is called instantiation.)

Advantages of OOP

- Ease of modeling real world in software context
- Object-oriented systems can be easily upgraded from small to large scale.
- It is easy to partition the work in a project based on objects.
- Object-oriented programming offers a new and powerful model for writing computer software.
- It reduces software maintenance and developing costs.
Etc...



IntelliJ IDEA

- <https://account.jetbrains.com/login>



- <https://www.jetbrains.com/help/idea/getting-started.html>

Summary

- What is Object Oriented Programming
- Fundamentals of Object Orientation
- Why Object Orientation ?
 - Modularity
 - Information-hiding
 - Code re-use
 - Pluggability and debugging ease
- Understanding Classes and Objects
- Real-World Scenario
- Class
- Object
- Instance
- Instantiation

References

- <https://docs.oracle.com/javase/tutorial/java/concepts/index.html>
- How To Program (Early Objects)
 - By H .Deitel and P. Deitel
- Headfirst Java
 - By Kathy Sierra and Bert Bates

Questions ???





Thank You