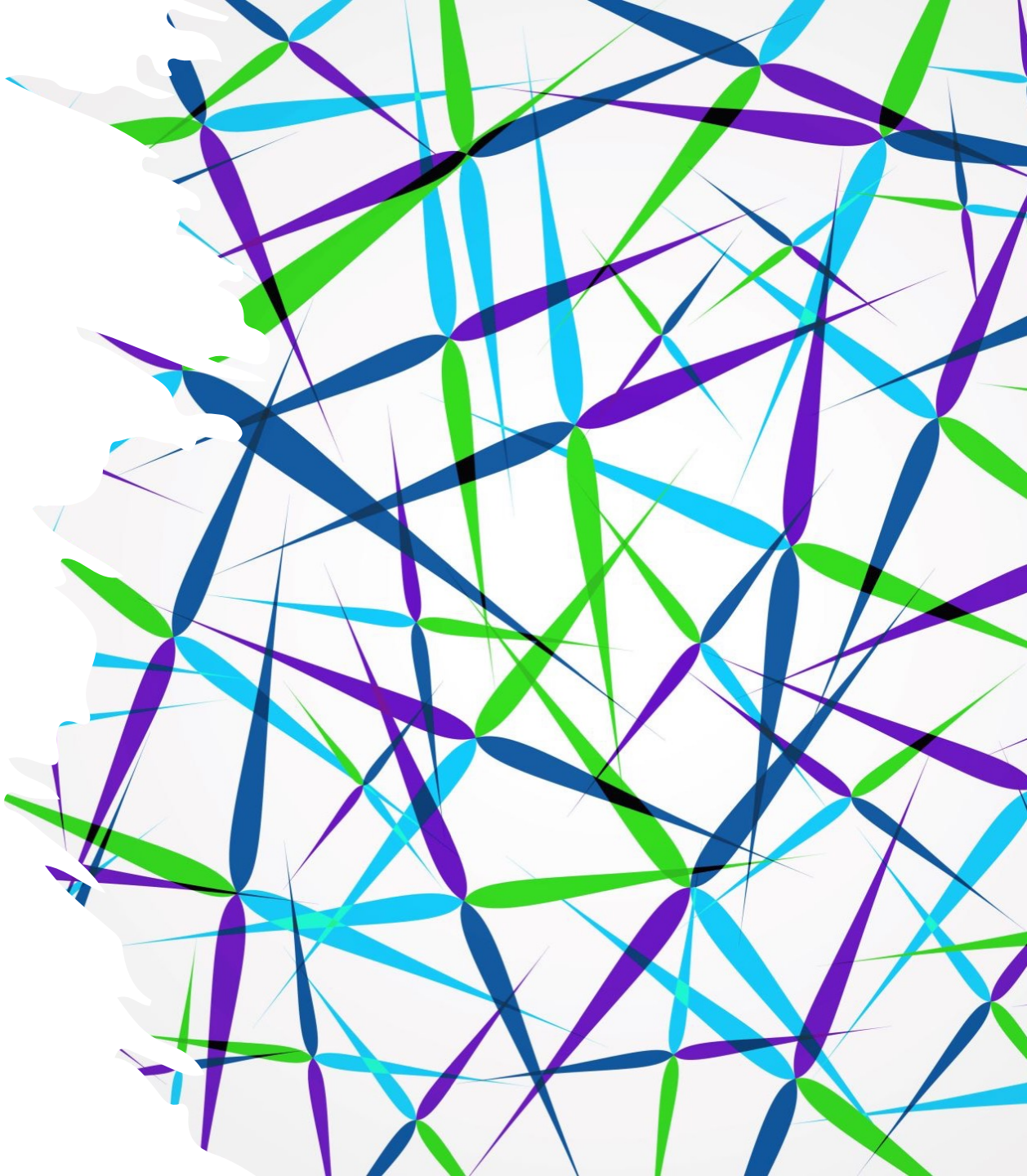
L2_OOPD_Java

PUSHPENDRA SINGH



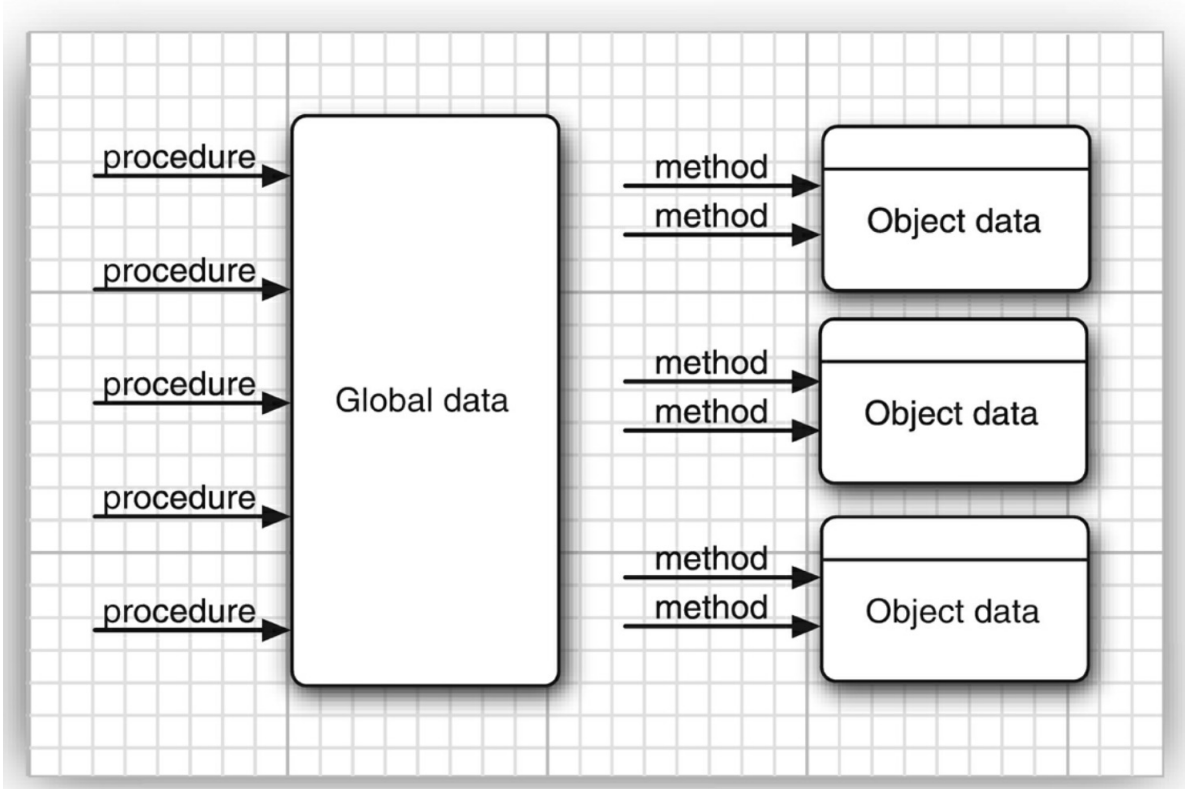
Procedural programming

- Structured programming consists of designing a set of procedures (or algorithms) to solve a problem.
- Once the procedures are determined, the traditional next step was to find appropriate ways to store the data.
  - Algorithms + Data Structures = Programs

Object-oriented programming

- An object-oriented program is made of objects.
- Each object has a specific functionality, exposed to its users, and a hidden implementation.
- Many objects in your programs will be taken "off-the-shelf" from a library; others will be custom-designed.
  - Whether you build an object or buy it might depend on your budget or time.
  - But, basically, as long as an object satisfies your specifications, you don't care how the functionality is implemented.

Procedural vs Object-oriented



Objects and Classes

- A class specifies how objects are made.
  - Think of classes as cookie cutters; objects are the cookies themselves.
- When you construct an object from a class, you are said to have created an instance of the class.

- Horstmann, Cay S.. Core Java, Volume I (p. 241). Pearson Education. Kindle Edition.

Source: MULTI Silver Stainless Steel Cookie Cutter Set For Kitchen Ware, Rs 45/piece | ID: 22200499262 (indiamart.com)

Objects and Classes

- All code that you write in an OOP like Java is inside a class.
- The standard language library supplies several thousand classes for such diverse purposes as user interface design, dates and calendars, and network programming.
- Nonetheless you still have to create your own classes to describe the objects of your application's problem domain.

Encapsulation

- Encapsulation (sometimes called information hiding) is a key concept in working with objects.
- Formally, encapsulation is simply combining data and behavior in one package and hiding the implementation details from the users of the object.
- The bits of data in an object are called its instance fields, and the procedures that operate on the data are called its methods.
- A specific object that is an instance of a class will have specific values of its instance fields. The set of those values is the current state of the object. Whenever you invoke a method on an object, its state may change.

Encapsulation

- The key to making encapsulation work is to have methods never directly access instance fields in a class other than their own.
- Programs should interact with object data only through the object's methods.
- Encapsulation is the way to give an object its "black box" behavior, which is the key to reuse and reliability.
- This means a class may totally change how it stores its data, but as long as it continues to use the same methods to manipulate the data, no other object will know or care.

Classes

- When you start writing your own classes, another tenet of OOP will make this easier: Classes can be built by extending other classes.
- For example, Java, comes with a “cosmic superclass” called Object. All other classes extend this class.
- When you extend an existing class, the new class has all the properties and methods of the class that you extend.
- You then supply new methods and instance fields that apply to your new class only. The concept of extending a class to obtain another class is called inheritance.

Objects

- Three key characteristics of objects:
  - Behavior: what can you do with this object, or what methods can you apply to it?
  - State: how does the object react when you invoke those methods?
  - Identity: how is the object distinguished from others that may have the same behavior and state?
- All objects that are instances of the same class share a family resemblance by supporting the same behavior.
- The behavior of an object is defined by the methods that you can call.

Objects

- Each object stores information about what it currently looks like. This is the object's state.
- An object's state may change over time, but not spontaneously. A change in the state of an object must be a consequence of method calls.
  - If an object's state changed without a method call on that object, someone broke encapsulation.
- However, the state of an object does not completely describe it, because each object has a distinct identity.
  - For example, in an order processing system, two orders are distinct even if they request identical items.
- Notice that the individual objects that are instances of a class always differ in their identity and usually differ in their state.
- These key characteristics can influence each other.
  - For example, the state of an object can influence its behavior.
    - If an order is "shipped" or "paid," it may reject a method call that asks it to add or remove items. Conversely, if an order is "empty"—that is, no items have yet been ordered—it should not allow itself to be shipped.

Starting OOP Programming

- In a traditional procedural program, you start the process at the top, with the main function.
- When designing an object-oriented system, there is no "top."

Starting OOP Programming

- Identifying Classes and Methods
  - Classes are “noun”
  - Methods are “verb”
- Relationships
  - Dependence (“uses–a”)
  - Aggregation (“has–a”)
  - Inheritance (“is–a”)

Reference

- Horstmann, Cay S.. Core Java, Volume I (p. 241). Pearson Education. Kindle Edition.