OBJECT ORIENTED PROGRAMMING

INDEX
UNIT 1 PPT SLIDES

Need for OOP Paradigm

- OOP is an approach to program organization and development, which attempts to eliminate some of the drawbacks of conventional programming methods by incorporating the best of structured programming features with several new concepts.
- OOP allows us to decompose a problem into number of entities called objects and then build data and methods (functions) around these entities.
- The data of an object can be accessed only by the methods associated with the object.

Some of the Object-Oriented Paradigm are:

1. Emphasis is on data rather than procedure.
2. Programs are divided into objects.
3. Data Structures are designed such that they Characterize the objects.
4. Methods that operate on the data of an object are tied together in the data structure.
5. Data is hidden and can not be accessed by external functions.
6. Objects may communicate with each other through methods.

A way of viewing world – Agents

- OOP uses an approach of treating a real world agent as an object.
- Object-oriented programming organizes a program around its data (that is, objects) and a set of well-defined interfaces to that data.
- An object-oriented program can be characterized as *data controlling access to code* by switching the controlling entity to data.

Responsibility

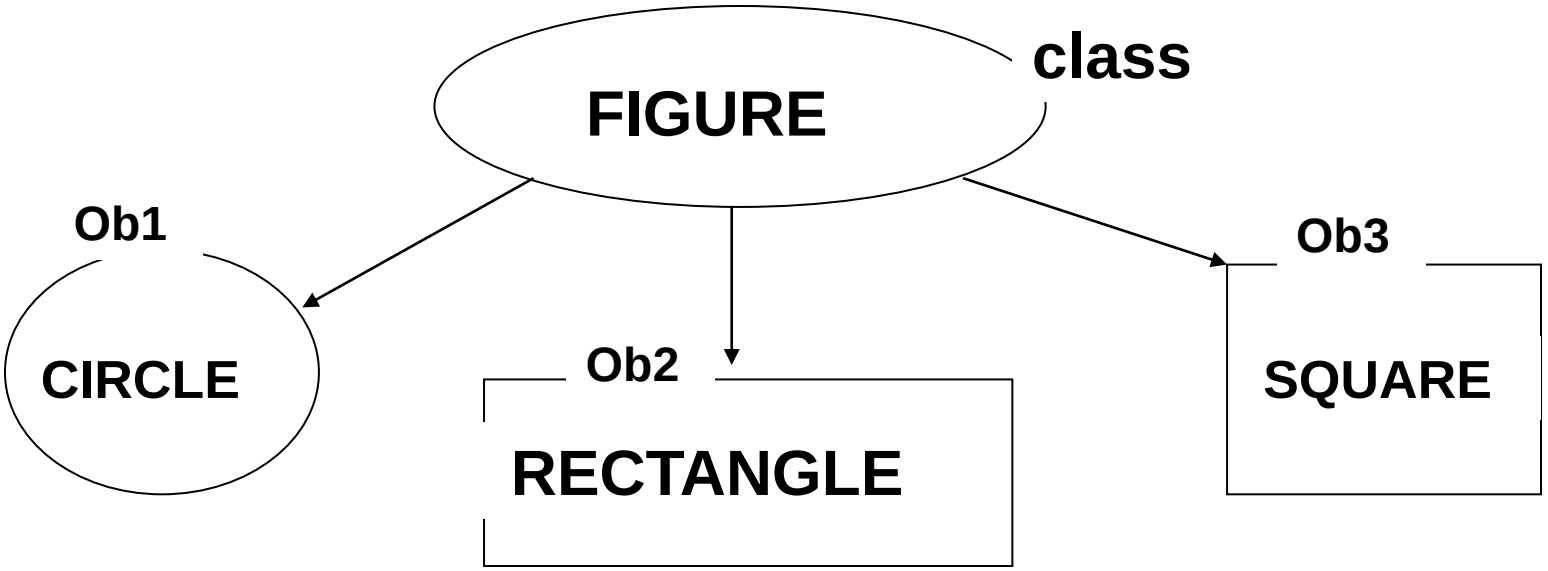
- primary motivation is the need for a platform-independent (that is, architecture- neutral) language that could be used to create software to be embedded in various consumer electronic devices, such as microwave ovens and remote controls.
- Objects with clear responsibilities
- Each class should have a clear responsibility.
- If you can't state the purpose of a class in a single, clear sentence, then perhaps your class structure needs some thought.

Methods

- A method is a group of instructions that is given a name and can be called up at any point in a program simply by quoting that name.
- Drawing a Triangle require draw of three straight lines. This instruction three times to draw a simple triangle.
- We can define a method to call this instruction three times and draw the triangle(i.e. create a method drawLine() to draw lines and this method is called repeatedly to achieve the needed task)
- The idea of methods appears in all programming languages, although sometimes it goes under the name *functions* and sometimes under the name *procedures*.
- The name *methods* is a throw-back to the language C++, from which Java was developed.
- In C++, there is an object called a *class* which can contain methods. However, everything in Java is enclosed within a class .so the functions within it are called methods

CLASSES

- Class is blue print or an idea of an Object
- From One class any number of Instances can be created
- It is an encapsulation of attributes and methods



syntax of CLASS

```
class <ClassName>
{
    attributes/variables;
    Constructors();
    methods();
}
```

INSTANCE

- **Instance is an Object of a class which is an entity with its own attribute values and methods.**

- **Creating an Instance**

```
ClassName refVariable;
refVariable = new Constructor();
   or
ClassName  refVariable = new  Constructor();
```

Java Class Hierarchy

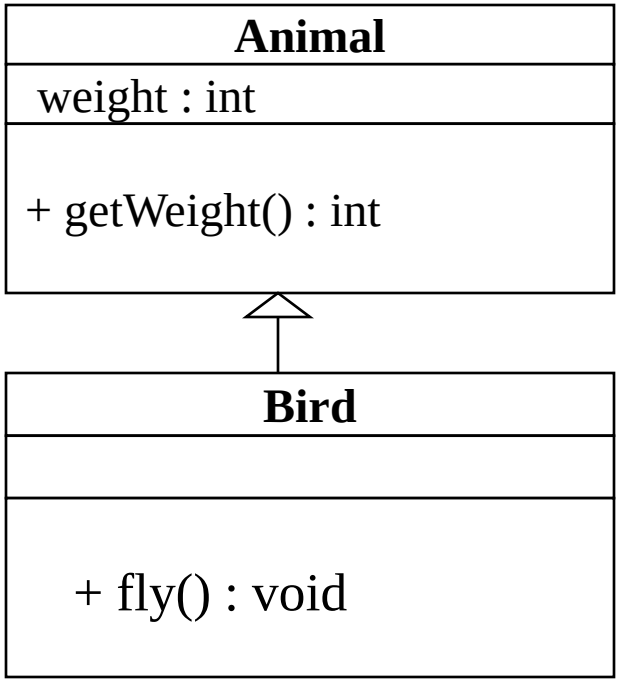
- In Java, class “Object” is the base class to all other classes
  - If we do not explicitly say extends in a new class definition, it implicitly extends Object
  - The tree of classes that extend from Object and all of its subclasses are is called the class hierarchy
  - All classes eventually lead back up to Object
  - This will enable consistent access of objects of different classes.

Inheritance

- Methods allows to reuse a sequence of statements
- *Inheritance* allows to reuse classes by deriving a new class from an existing one
- The existing class is called the *parent class,* or *superclass*, or *base class*
- The derived class is called the *child class* or *subclass*.
- The child class inherits characteristics of the parent class(i.e the child class *inherits* the methods and data defined for the parent class

Inheritance

- Inheritance relationships are often shown graphically in a *class diagram*, with the arrow pointing to the parent class



Method Binding

- Objects are used to call methods.
- **MethodBinding** is an object that can be used to call an arbitrary public method, on an instance that is acquired by evaluatng the leading portion of a method binding expression via a value binding.
- It is legal for a class to have two or more methods with the same name.
- Java has to be able to uniquely associate the invocation of a method with its definition relying on the number and types of arguments.
- Therefore the same-named methods must be distinguished:

  1) by the number of arguments, or

  2) by the types of arguments
- Overloading and inheritance are two ways to implement polymorphism.

Method Overriding.

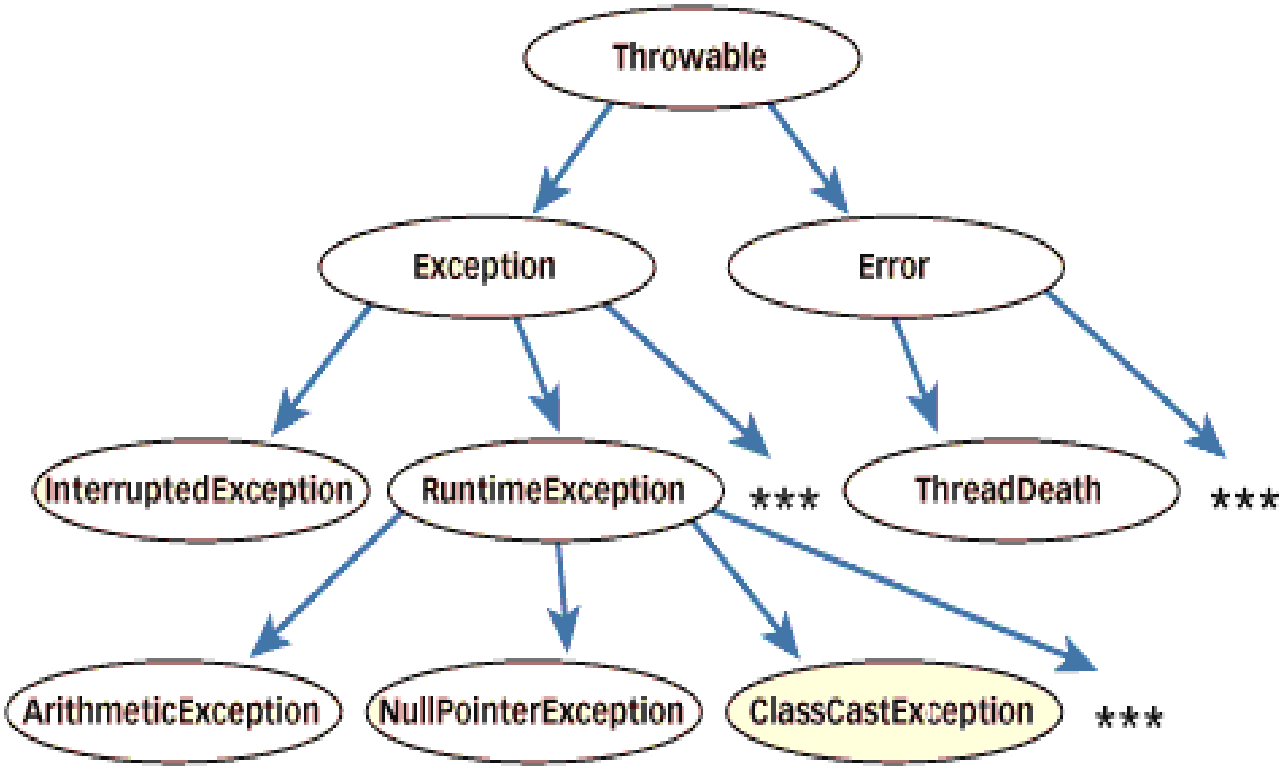
- There may be some occasions when we want an object to respond to the same method but have different behaviour when that method is called.
- That means, we should override the method defined in the superclass. This is possible by defining a method in a sub class that has the same name, same arguments and same return type as a method in the superclass.
- Then when that method is called, the method defined in the sub class is invoked and executed instead of the one in the superclass. This is known as overriding.

Exceptions in Java

- Exception is an abnormal condition that arises in the code sequence.
- Exceptions occur during compile time or run time.
- “throwable” is the super class in exception hierarchy.
- Compile time errors occurs due to incorrect syntax.
- Run-time errors happen when
  - User enters incorrect input
  - Resource is not available (ex. file)
  - Logic error (bug) that was not fixed

Exception classes

- In Java, exceptions are objects. When you throw an exception, you throw an object. You can't throw just any object as an exception, however -- only those objects whose classes descend from Throwable.
- Throwable serves as the base class for an entire family of classes, declared in java.lang, that your program can instantiate and throw.
- Throwable has two direct subclasses, Exception and Error.
- Exceptions are thrown to signal abnormal conditions that can often be handled by some catcher, though it's possible they may not be caught and therefore could result in a dead thread.
- Errors are usually thrown for more serious problems, such as OutOfMemoryError, that may not be so easy to handle. In general, code you write should throw only exceptions, not errors.
- Errors are usually thrown by the methods of the Java API, or by the Java virtual machine itself.


Throwable
Exception
Error
InterruptedException
RuntimeException
***
ThreadDeath
***
ArithmeticException
NullPointerException
ClassCastException
***

Summary of OOPS

The following are the basic oops concepts:

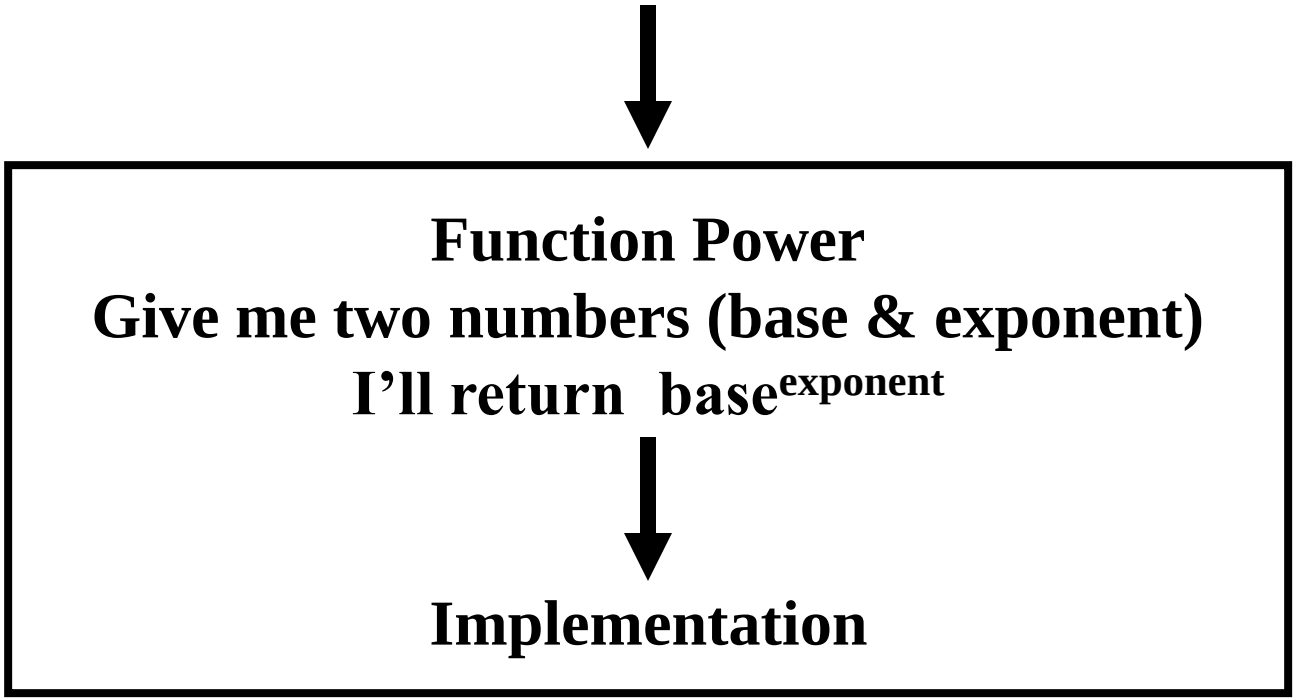
They are as follows:

1. Objects.
2. Classes.
3. Data Abstraction.
4. Data Encapsulation.
5. Inheritance.
6. Polymorphism.
7. Dynamic Binding.
8. Message Passing.

Abstraction in Object-Oriented Programming

Procedural Abstraction

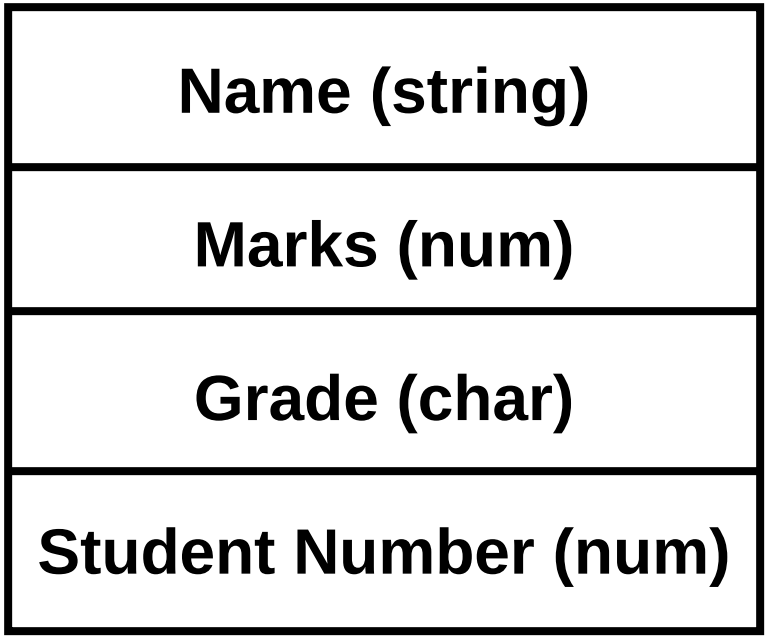
- Procedural Abstractions organize instructions.



Data Abstraction

- Data Abstractions organize data.

StudentType



Behavioral Abstraction

- Behavioral Abstractions combine procedural and data abstractions.