UNIT-II: IMPERATIVE PARADIGM: DATA ABSTRACTION IN OBJECT ORIENTATION

Faculty In-charge

Aaysha Shaikh

Assistant Professor (IT Dept.)

email: aayshashaikh@sfit.ac.in

OF UNIT-2

OUTLINE

✓ 2.1 Grouping of data and operations

✓ 2.2 Encapsulation

2.3 2.4 2.5 2.6

Overloading and

polymorphism

Inheritance

Initialization and finalization

Dynamic Binding

2.1:-Grouping of data and operations

LETS REVISE FIRST……

OBJECT ORIENTED PROGRAMMING PARADIGM

. An approach to the solution of problems in which all computations are performed in the context of objects

. The program is written as a collection of classes and objects.

. The smallest and the basic entity is object

. Emphasis is on data rather than procedure

. Methods that operate on the data of an object is tied together in the data structure

. Data is hidden and cannot be accessed by external function

. Objects may communicate with each other through methods

. Follows bottom up approach in program design

. Ruby, Java, C++, Python, Simula

SOME KEY POINTS

1. Development in software technology continue to be dynamic process

2. Most programming languages tries to answer-

. How to represent real life entities of problem in system design

. How to design systems with open interfaces

. How to ensure reusability and extensibility of modules

. How to develop modules that are tolerant to any change in future .

How to improve software productivity and decrease software cost .

How to improve the quality of software

Tobuild todayscomplexsoftwareweneedto–
Incorporatesoundconstructiontechniquesandprogramstructures
that are easyto comprehend,implementandmodify.
Weneedto moveaheadfromputtingtogethera set ofprogrammingstatements

STRUCTURE OF PROCEDURE ORIENTED PROGRAMMING

. Number of functions are written to accomplish any task

. Primary focus is on functions. Functions share global data

. Global data are more vulnerable to an inadvertent change by a function . In large program it is very difficult to identify what data is used by which function

. In case of revision, we need to revise all functions that access the data
. Fails to model real world problems

PROBLEM

- OOP allows decomposition of problem into a number of entities called objects
- Data and functions are build around the objects
- The data of an object can only be accessed only by the functions associated by the objects

. Functions of one object can access the functions of other objects

FEATURES OF OBJECT ORIENTED PROGRAMMING CONCEPTS

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

OBJECTS CLASSES

4. Functions that operate on the data of an object are **tied** together in the data structure

DATA ENCAPSULATION

5. Data is hidden and cannot be accessed by external functions

ABSTRACTION

POLYMO

INHERITA

RPHISM 6. Objects may **communicate** with each other

NCE

through functions

7. New data and functions can easily be **added** whenever necessary

OBJECTS

. Objects are basic run-time entities

. They may represent a person, a place, a bank account, a table of data…..or any other item

Objects take up space in the memory and have an associated address

CLASSES

. Objects contain data and code to manipulate the data

. In-fact objects are variables of type class

. Once a class is created we can create any number of variables of that class .

Each object is associated with the data of type class with which they are created . Aclass is thus collection of objects of similar type

Example: mango, apple orange are members of class fruit

Example of a Class

Class: Course

✓ Properties: Name, Location, Days Offered, Credit Hours, Professor ✓
Behavior:Add Student, Delete Student, Get Course Roster, Determine If Full .
AClass is a description of a group of objects with common properties (attributes), behavior (operations), relationships, and semantics

. Aclass is an abstraction.An object is an instance of a class

Attribute

. An Attribute is a named property of a class. It has a type. It describes the range of values that that property may hold.

Operation (Function)

. An Operation is a service that can be requested from any object of the Class to affect behavior.

. An Operation can either be a command or a question.

2.2:-Encapsulation (C++, Java , Pyhon)

Assume, I havea project…..
The project getsdividedintolotsof subprojects

Class1
(object1)

Class2
(object2)

Class3
(object3)

Each is

a
single
unit

Data Encapsulation : It is a process of combining data members and functions in a singleunit called class

. Default: When no access modifier is specified for a class , method or data member –It is saidto be havingthe default accessmodifier by default.

. The data members, class or methods which are not declared using any access modifiers i.e. having default access modifier are accessible only within the same package

One class one object

One class more object

more class more object

Initializing using functions (static)

Initializing using functions (dynamic)

Initializing using constructor

Has same name as the class

Creation of Packages

1. Save the file as demo.java

Compile…..InmycaseIhadgotfewerrors

Successful compilation

Create a package

Compile file within package

Run file within package

DATA ABSTRACTION

. **You need to know**

How to use the coffee machine to make coffee

Provide water and coffee beans, switch it on and select the kind of coffee you want to get.

The thing you don't need to know is

✓

How the coffee machine is working internally to brew a fresh cup ✓ The ideal temperature of the water, amount of ground coffee, sugar added .

Someoneelseworried about that and created a coffeemachine that nowactsasanabstractioandhidesallthese details

DATA ABSTRACTION

. Objects in an OOP language provide an abstraction that hides the internal implementation details.

. Similar to the coffee machine in your kitchen, you just need to know

which methods of the object are available to call and which input parameters are needed to trigger a specific operation.

. **But you don't need to understand how this method is implemented and which kinds of actions it has to perform to create the expected result**.

. **Definition:** Data abstraction is a process to model/create our own user defined data types (using class and constructs) and then define variables (objects of those new data types)

. Is a simplified view of an object that includes only features one is interested in while hides away the unnecessary details

. Data abstraction becomes anAbstract Data Type (ADT) or user defined type

DATAABSTRACTION in C++

. Data abstraction in C++ is achieved through classes

```
#include <iostream> using namespace std;
class implementAbstraction {
private: int a, b;
public:
        // method to set values of private members
            void set(int x, int y) {
            a = x;
            b = y;
            }
            void display() {
                    cout<<"a = " <<a << endl;
                    cout<<"b = " << b << endl;
            }
};
```

Members declared as **public** in a class, can be accessed from anywhere in the program.

Members declared as **private** in a class, can be accessed only from within the class. They are not allowed to be accessed from any part of code outside the class..

public members can access the private members as they are inside the class.

```
int main()
{
            implementAbstraction obj;
            obj.set(10, 20);
            obj.display();
```

Constructor has same name as

the class itself. class construct {

public:

int a, b;. **Constructors** don't **have** return // Default Constructor type.

construct()

{. A**constructor** is automatically

a = 10; b = 20;

called when an object is **created**.

}. If we do not specify

};

a **constructor**, **C++** compiler

int main()

generates a

```
{ default constructor for us
   // Default constructor called
   automatically // when the object is
   created
   construct c;
   cout << "a: " << c.a << endl
        << "b: " << c.b;
   return 1;
}
```

an empty body).

(expects no parameters and has

DATAABSTRACTION in Java

. Data abstraction in java is achieved through **interfaces and abstract classes** .

The abstract keyword is a non-access modifier, used for classes and methods:

. **Abstract class:** is a restricted class that cannot be used to create objects (to access it, it must be inherited from another class).

. **Abstract method:** can only be used in an abstract class, and it does not have a body. The body is provided by the subclass (inherited from).

. An abstract class can have both abstract and regular methods:

```
abstract classAnimal { Animal myObj = new Animal(); // will generate an error public abstract
 void animalSound();
public void sleep()
 {(will introduce error)
System.out.println("Zzz");
}
 }
```

DATAABSTRACTION in Java-Rules

An abstract class must be declared with an abstract

Abstract classes can have abstract and non-abstrac t

It cannot be instantiated. (object cannot be created directly keyword. using newkeyword) methods

It can have final

It can have constructors and static methods also.

methods which will force the subclass not to change the body of the method.

Declaration of abstract class and

methods

Inheritance of abstract class

Creation of object of inherited class

- Another way to achieve abstraction in Java, is with interfaces.
- An interface is a completely "**abstract class**" that is used to group related methods with empty bodies:

. To access the interface methods, the interface must be "implemented" by another class with the implements keyword (instead of extends)

. The body of the interface method is provided by the "implement" class:

. Like **abstract classes**, interfaces **cannot** be used to create objects (in the example above, it is not possible to create an "Animal" object)

. On implementation of an interface, you must override all of its methods

. Interface methods are by default abstract and public
. Interface attributes are by default public, static and final

Declaration of

interface and its methods

Extending the interface

Creating object of extended class

Declaration of first and second interfaces

Extending the interfaces

Creating object of extended class

. AJava class can implement multiple Java Interfaces. It is necessary that the class must implement all the methods declared in the interfaces.

. Class should override all the abstract methods declared in the interface

. All methods in an interface are implicitly public and abstract . An

interface cannot be instantiated

. An interface which is declared inside another interface is referred as nested interface

. The class cannot implement two interfaces in java that have methods with same name but different return type.

Abstraction is the method of hiding Encapsulation is a method to hide the data the
unwanted information. in a single entity or unit along with a method to protect
information from

outside.

We can implement abstraction using abstract class and interfaces.

Whereas encapsulation can be implemented using by access modifier i.e. private, protected and public.

In abstraction, implementation complexities are hidden using abstract classes and interfaces.

encapsulation, the data is hidden using methods

References

1. Michael L Scott, “ Programming Language Pragmatics”, Third edition, Elsevier publication (Chapter-9, specifically 9.1 and 9.2)

2. Ravi Sethi, “ Programming Languages-concepts and constructs”, Pearson Education

(Chapter-6)

3. NPTEL lecture series on Programming in Java, IIT Kharagpur
https://www.youtube.com/watch?v=K9gQwLeNXyw&list=PLbRMhDVUMngcx5xHChJf7ofxZI4JzuQR&index=8