ADS using Java

Prof. Pujashree Vidap

PICT, Pune

psvidap@pict.edu

Group G

- Implement the Heap/Shell sort algorithm implemented in Java demonstrating heap/shell data structure with modularity of programming language.
- Any application defining scope of Formal parameter, Global parameter, Local parameter accessing mechanism and also relevance to private, public and protected access. Write a Java program which demonstrates the scope rules of the programming mechanism.
- Write a Java program which will demonstrate a concept of Interfaces and packages: In this assignment design and use of customized interfaces and packages for a specific application are expected.
- Write a Java program which will demonstrate a concept of cohesion and coupling of the various modules in the program.

- Write a program on template and exception handling in Java: in this assignment multiple templates are to be designed as a pattern and these patterns to be used to take decisions.
- Write a Java program for the implementation of different data structures using JAVA collection libraries (Standard toolkit library): at least 5 data structures are used to design a suitable application.
- Design a mini project using JAVA which will use the different data structure with or without Java collection library and show the use of specific data structure on the efficiency (performance) of the code.

Assignment No:11

- Write a Java program which will demonstrate a concept of Interfaces and packages: In this assignment design and use of customized interfaces and packages for a specific application are expected.

- Example:
- **Write a java program in which interface CommonList is define in package common with methods like add, remove, size, isempty, display. A class SLL in define in package specificList implements CommonList. Implements all methods given in an interface CommonList in SLL class.**

Interface

- Blue print of class
- A mechanism to achieve abstraction
- An interface declares (describes) methods but does not supply bodies for them

```
interface KeyListener {
    public void keyPressed(KeyEvent e);
    public void keyReleased(KeyEvent e);
    public void keyTyped(KeyEvent e);
}
```

- All the methods are implicitly public and abstract
  - You can add these qualifiers if you like, but why bother?
- You cannot instantiate an interface
  - An interface is like a *very* abstract class—*none* of its methods are defined
- An interface may also contain constants (final variables)

- You extend a class, but you implement an interface
- A class can only extend (subclass) one other class, but it can implement as many interfaces as you like
- Example:

```
class MyListener
        implements KeyListener,
ActionListener { … }
```

Points to Remember

- Each method in an interface is also implicitly abstract, so the abstract keyword is not needed.
- Methods in an interface are implicitly public.
- You cannot instantiate an interface
- An interface is like a *very* abstract class—*none* of its methods are defined
- An interface may also contain constants (final variables)

Packages

- Containers for classes to avoid name collision
- Stored in hierarchical manner and explicitly imported into new class using import statement.
- Both naming and visibility control mechanism (Access Protection)
- Defining packages
- Importing packages

Assignment 12:

- Write a program on template and exception handling in Java: in this assignment multiple templates are to be designed as a pattern and these patterns to be used to take decisions.
- Example:

  Write a program for implementation of stack ADT in java, Stack is abstract base class with template method consists of push and display. IntergerStack and CharStack are two concrete classes inherited from Stack which give specific implementation of these methods. Also handle stack full and stack empty condition as exceptions

Points to remember

- Design patterns are the best practices used by experienced object-oriented software developers
- 23 design patterns are classified into Creational, Structural and Behavioral patterns
- Behavioral patterns are specifically concerned with communication between objects.
- Template method pattern is a behavioral design pattern which provides base method for algorithm, called template method which defers some of its steps to subclasses
- Template method which consists of steps which can be abstract method which will be implemented by its subclasses

```
abstract class Generalization {
   public final void findSolution() {
      stepOne();
      stepTwo();
      stepThr();
      stepFor();
   }
   protected void stepOne() { //Generalization stepOne }
   abstract protected void stepTwo();
   abstract protected void stepThr();
   protected void stepFor() { //Generalization stepFor  }
}

class Specialization extends Generalization {
   protected void stepThr() {
      // Specific implementation of method for class Specialization
   }

   protected void stepTwo() { // Specific implementation of method for class
                                Specialization}
}
```

- Generalization algorithm = new Specialization ();
- algorithm.findSolution();

Exception Handling

- An *exception* is an error condition that changes the normal flow of control in a program
- When something unexpected occurs
  - Ensure program detects the problem
  - Then program must do something about it
- Exceptions are runtime errors
- Exception handling gives us another opportunity to recover from the abnormality.
- Separates business logic and error handling code

Using throws clause

If you don't want the exception to be handled in the same function you can use the throws class to handle the exception in the calling function.

```
public class MyClass{
    public static void main(String args[]){
        try{
                checkEx();
        } catch(FileNotFoundException ex){
        }
    }
    public void checkEx() throws FileNotFoundException{
        File f = new File("myfile");
        FileInputStream fis = new FileInputStream(f);
        //continue processing here.
}
}
```

Assignment No:13

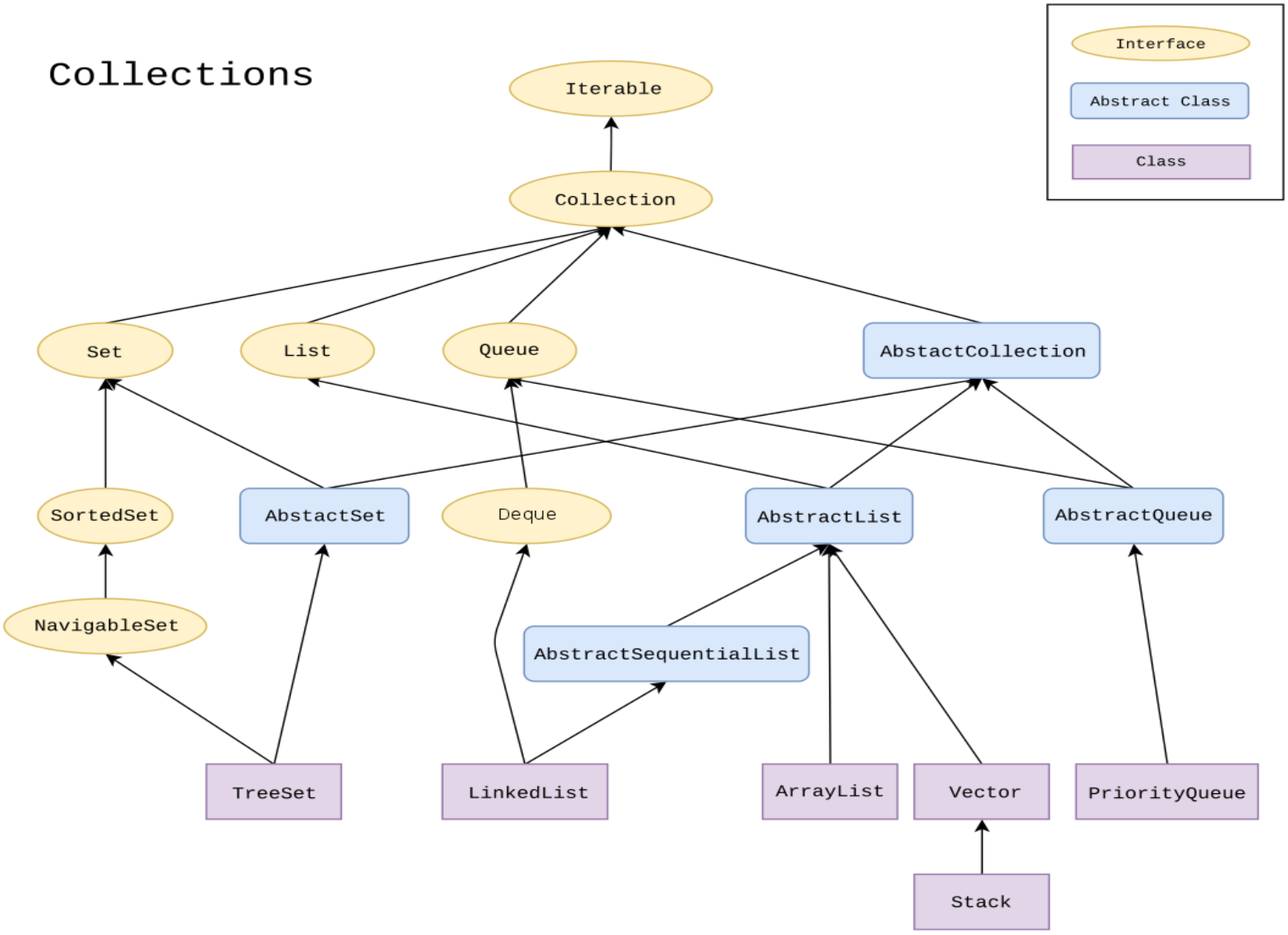
- Write a Java program for the implementation of different data structures using JAVA collection libraries (Standard toolkit library): at least 5 data structures are used to design a suitable application
- Example:

  Implement dictionary using different data structures in Java

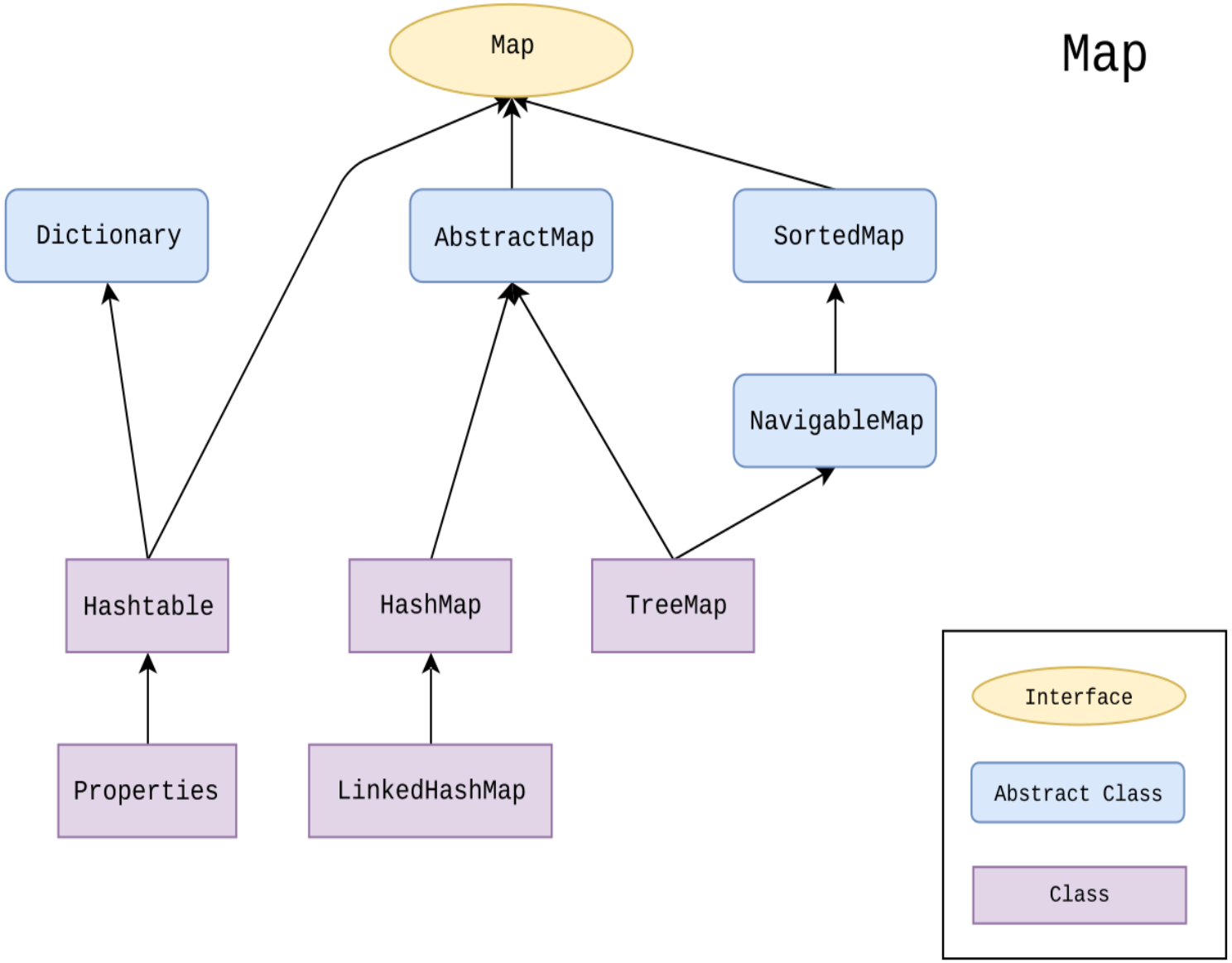
Introduction to Collection Framework

- Need of CF
- Collection : Group of individual objects represented by single entity
- Advantages
  - Grown able in nature , contains homo and heterogeneous objects
  - Every collections class implemented on DS so we can use readymade methods

Collections



Map


Map
Dictionary
AbstractMap
SortedMap
NavigableMap
Hashtable
HashMap
TreeMap
Properties
LinkedHashMap
Interface
Abstract Class
Class

Thank you

Questions ???