**Summary Keywords**

Program/Application….code that executes?

What is a class?...container of code…methods/attributes/data members/fields…..Encapsulation

Types of classes, User defined classes, API….Instantiable, Utility, GUI,IO

Examples of Classes: JOptionPane,String,Integer

Intro to package, Class (container for code), object?

Class header ( public class ClassName)

main method, method header……*public static void main(String [] args)*

other methods?

Method parts….Modifiers…Scope/returntype..static(Class vs Instance)…name

Scope(can apply to variables/classes/methods

Calling methods…..writing user defined methods

Some Class methods used: *parseInt(….), showMessageDialog(….), showInputDialog(…), format(….)*

Classname.method(..,…,…) e.g. *JOptionPane.showMessageDialog(null,“welcome”);*

Assignment statement……*name =JOptionPane.showInputDialog(“Enter Name”)*

Java API

Variables vs Arguments vs Parameters

Recap of conditional and looping structures (*If….Switch…For….While….Do..While*)

{ } brackets (start and finish things)

Data Types …*String*….primitive data types…..*int, float, double*

Declaring variables…..*int age;*

Naming Convention……lowercase, Uppercase, camelCase

**What is an object?**

Java API

Method header…modifiers, returntype, parameter type, parameter.

*static* keyword

User defined methods….how to call them, passing arguments (pass by value)

Scope *public, private* (can apply to variables, methods, classes)

User defined class………Example

Programming as manipulation of objects/use of class functionality

objects….instance of a class? *new* keyword..instantiation/creation….memory

Abstraction, capture only those details about an object that are relevant to the current perspective

Class as a blueprint

Attributes…instance methods, instance methods versus class methods

Instantiable Class (Attributes/methods)

Types of methods

Constructors

Accessor/Mutator methods

Github

Different kinds of classes --- Encapsulation, information hiding, public interface

Types of Classes

--- Utility classes ---- Collection classes--- Math

--- Wrapper classes

--- GUI/graphics classes

--- Instantiable classes

---- I/O classes

--- Network classes

--- Exception classes

Class

--- methods—class,instance

--- attributes/instance variable/data member/fields/class variables/constants

UML --Person class, try another ---Start from UML class diagram

True encapsulation

Multi Argument Constructors

Method overloading

Scope of attributes/methods

Class methods attributes versus instance methods and attributes

Keyword *static* static methods/variables

View of Participating classes/ UML class diagrams

Keyword *this*

Test classes

Array of objects…creating, populating processing and displaying

Polymorphism

Object class

Github - version control

Code referencing - see doc on referencing

Four basic principles of OOP- Encapsulation, Abstractions, Inheritance, Polymorphism

Driver classes/Test classes/ Junit Testing

Javadoc – see customer example

Creating an instance of a class within the class definition,

compile time versus runtime

Aggregation/Composition ‘has a’ relationship

Hierarchy/Class/VOPC diagram

Method overloading/overriding

Inheritance ‘is a’ relationship

Superclass/Subclass

Interface

Interfaces, SuperClass/SubClass, Hierarchy, Objects

Keywords: extends, implements, super

Abstract classes

Abstract methods

Event Driven Programming

Model: Source, Event, Handler/Listener, Registering, firing events, handling events

Interfaces: WindowListener/ActionListener

Classes: JFrame, ActionEvent,

Methods: addActionListener, actionPerformed, getSource, getActionCommand

Inner Classes

Anonymous Inner Classes

Menu Systems

Creating a menu, JMenuBar, JMenu, JMenuItem,

Processing Collections(Array, Arraylist, LinkedList) of Objects

Creating an array of objects

Enhanced for loops

Full System(Bicycle system for sample programs)

Event Driven Programming

Model: Source, Event, Handler/Listener, Registering, firing events, handling events

Interfaces: WindowListener/ActionListener

Classes: JFrame, ActionEvent,

Methods: addActionListener, actionPerformed, getSource,getActionCommand

Streams

Persistence

Low level (Bytes/Characters)/high level I/O(Objects)

Opening / Saving files

Exceptions

Try….catch….finally

Propagation of exceptions….throws

Creation of exceptions….throw

Clean Code

Secure Code

Annotations