Week 1 – Computer Basics, Memories, Unix & Core JAVA

Day 1:

OS Fundamentals –

* Operating System (OS) is a System Software
* With the help of OS, any one can interact with the hardware of the System
* CUI OS – Character User Interface (Command) Based Operating System
* GUI OS – Graphical User Interface (Click) based Operating System
* CUI OS – MS-DOS, Unix
* GUI OS – Windows, Linux, Mac, Solaris (Java lang Based OS)
* Multi-User, Multi Tasking Operating System

Unix Commands

* Ls – List file and directories
* Whoami – print the current user details
* Users – display user info
* Mkdir – creating a folder
* Cd – changing to a directory/folder
* Touch – To create a new file
* Cp – copying file/folder
* Chmod – change mode command – to change the accessibility of file or folder
* Date
* Time
* Cal

SDLC – Software Development Life Cycle

Phases of SDLC (WaterFall Model)

* Requirement Gathering (Client Meetings)
* Analysis
* Design
* Coding
* Testing
* Delivery
* Maintenance

Agile Method of Delivery

Agile follows small and incremental process called Sprint which will deliver the working software at the end of each sprint

1. Agile Won’t encourage documentation works
2. Agile follows very simple process
3. Team members can decide on the modules to work
4. Work will not be assigned by anyone rather it will be chosen by the team members
5. Agile follows very simple documents called Product Backlog, Sprint Backlog and Breakdown chart
6. In Agile, the requirements are gathered as Epics and User Stories
7. All the Epics/ User stories will be documented by the Scrum Master with the product owners knowledge
8. In Agile, Changes are welcomed all the time, even late in the development
9. Agile Ceremonies – Sprint Planning, Sprint review/retrospection, Daily Standup
10. Agile team – Scrum Master, Scrum Team
11. It measures the progress in terms of working code delivered at the end of each sprint

Day 2

Environment Variables – Setting a folder in the System variable so that it can be accessed from any folders.

Package Managers – yum (Yellowdog Update Manager), rpm (RedHat Package Manager), APT (Adv Packing Tool)

File Permissions – Chmod command

Day 3 Core JAVA

Introduction to JAVA

Pillers of OOP

Class VS Object

JDK, JRE, JVM

Method Parameters

* Java is a very popular, multi-threaded, Object oriented, high level, platform indenpendent and highly secured, general purpose programming lang
* Java is a compiled & interpreted programming lang
* JDK – Java Development Kit (Which will contains tools to convert the source code to byte code .java -🡪 .class)
* JRE – Java Runtime Environment (Which will help to run the Java related applications on JVM)
* JVM – Java Virtual Machine – It’s a computer which is completely created by software only
* Java is platform independent lang but JDK, JRE, JVM all are platform dependent
* Java is not a Pure Object Oriented programming lang bcos of primitives
* Java will not allow explicit pointer manipulation
* Java will not support multiple polymorphism (A class can’t extend more than one class)
* To invoke the JVM from command prompt adding a new Environment variable namely “JAVA\_HOME” which is pointing to the location where JDK/JRE/JVM is installed.
* JAVA\_HOME needs to be updated in path environment variable also
* Class VS Object [ Class is a blue-print to create objects – Class is a specification]
* Object – [Instance of a class – Implementation of a class]
* Using a single class, many objects can be created
* Writing a Simple HelloWorld Java program .
* Method Parameters
* Class will define both properties (State) & methods (Behaviors)
* OOP features – A PIE (Abstraction, Polymorphism, Inheritance, E – Encapsulation)
* Abstraction – Hiding the implementation (Showing relevant data and hiding irrelevant date)
* Abstraction is achieved using Abstract class and interface
* Polymorphism – (Static/Compile Time, Dynamic/ Run-time Polymorphism)
* Polymorphism is achieved with the help of Method Overloading & Overriding
* Method overloading (Static or Compile time), redefining same method with different signature
* Method Overriding (Dynamic or Runtime polymorphism), redefining a parent class method in child class without changing the signature of the method.
* Inheritance – Getting properties of other classes or interface.
* Single Inheritance/Multiple/Multi – level inheritance
* Inheritance can be achieve using extends keyword
* Encapsulation (Hiding the Data ) – Making properties/ state as private & making methods/behaviors are pulic – so that private members can be accessed using public methods.

Day 4

* Constructor – Is a special method which will have the same name as of class.
* Constructor will not return anything even void also no need to add while defining a constructor
* Constructor is used to initialize all the member variables and also to create new object in heap memory area
* Packages – Are used to avoid naming conflict. It uses reverse of company url.
* Package name should be small case only.
* Examples : com. revature, com.google.api
* Wrapper Classes – These are built-in classes which helps to convert primitive data type to objects.
* There are 8 wrapper classes to support 8 different data tpes.
* Git – Open Source distributed Command/Click based version control system.
* Git Commands using Gitbash and Cygwin
* Git init, git add ., git commit -m “commit message”, git log, git status, git log oneline
* Git pull, git push

Day 5

* Type Casting – converting from one primitive to another primitive data type
* Control Flow statements
* Age vs Waterfall

Types of type casting

1. Implicit Type Casting (boolean-byte, byte- char, char – short, short-int, int-float, float -double)
2. Explicit Type casting (It will have the target date type mentioned in the brackets
3. Control flow statement [Used to control the program flow]
4. Types of Control Flow statement
   1. Conditional Checking statements (if, else if ladder, switch)
   2. Looping / Repetitive statements
      1. Entry Control Loop (while & for)
      2. Exit Control Loop (do while)

Week2

Day 6

Access Modifiers (private, package/default, protected, public) – which decide who/how the members can be accessed in a Java class)

Non-Access Modifiers (Behavior modifiers – static, abstract, final, synchronized, transient, volatile)

String API – Immutable Character Array (Source can’t be modified after creation, for each update it will create a new copy which will waste the memory )

StringBuffer & StringBuilder – Mutable arrays of character – which can be modified after creation

Arrays – It’s a derived data type. Which store similar kind of data in a continuous memory space

Var-Args – Variable Arguments. (It uses … ) [It indicates zero or more values/arguments can be passed]

Var-Args are used in method declaration/definition

Enhanced for loop (for each loops) – Which will not have 3 parts as normal for loop, [loop var, condition check, inc/dec operation) –

Enhanced for each loop needs a loop variable (normally will not be primitive) and group of data.

Variable Scopes – Types of variable [ Local variables, instance variables & static/class variables]

Local var will be accessed only with in the block where it is defined. – Variables declared inside the method and passed as an argument to the method

Instance variable – variable declared inside class & outside any method. It can be accessed by all the methods of the same class. [accessed modifiers will decide who else can access these variable outside class]

Static/class var – are Class level variables that can be accessed using the class name directly. A single copy will be shared by all the objects of the class

Day 7

Object Class – clone(), equals(), hashcode(), toString(),

Exception Handling – try/catch block (recommended way), using throws keyword (passing the Exception to the JVM)

Keyword used in exception handling – try, catch, finally, throw and throws

Throw keyword is used to throw an exception.

Throwable is the root of Exception hierarchy. (Exception/Error)

Types of Exception – Checked/Compile time exception [IOException, File,SQL], Un-checked/runtime exception [Arithmetic, IndexOutOfBound]

Custom Exception – Needs to extend either Exception or RuntimeException Class

Day 8

Unit Testing using Junit – TDD (Test Driven Development) – Unit Testing

Getting user input using Scanner – next(), nextInt(), nextLine()

Interfaces & Abstract Classes – Interface VS Abstract Class

Day 9

Annotations – Meta data (Data about data)

Collection API (List, Set, Queue)

Stack & Heap

Garbage Collection

Day 10

HashSet Vs TreeSet

ArrayList VS LinkedList

HashMap

ArrayDeque VS PriorityQueue

Iterable VS Iterator

Comparable VS Comparator