JAVA BASICS:

Example.java

Public class Example {

int age;

string name;

float salary;

// method/function

void display(String str){ // parameter

System.out.println(str);

}

Public static void main(String args[])

{

Example ex = new Example(); // Constructor

ex.display(“hi”); // argument

}

}

Data types:

Variables:

Access modifiers: private, public, protected and default

Parameters and the arguments

OOP:

Abstraction

Inheritance

Polymorphism

Encapsulation

Primitive and non primitive data types.

Collections: List, Set, Map…

Variable types:

Local Variables : Blocks {}, methods

Instance variables: class variables

Static variables: Global variable, not pertained to an object.

Switch:

Multiple If-else statements.

Switch(condition){

Case 1:

Break;

Case 2:

Break;

Case 3:

Break;

Default:

}

The value of a case should be a constant.

The condition inside the switch can be of any primitive type / enums / Strings.

Looping statements in Java:

For(), while(), do while()

5 employees and I want to print the ids of all the 5 employees

a[]

For(int i=0;i<5;i++){

s.o.p(a[i]);

}

Int i=0;

While(condition ex: i<5)

{

I++;

}

Do{

}while(condition check post execution of the code)

Use the break and continue statements in your looping code.

For(int i=0;i<5;i++){

If(i==0) continue;

s.o.p(i);

}

s.o.p(“outside loop”)

while()

do while()

Blocks:

Public static void main(String args[]){

Int x = 1;

s.o.p(x);

{

Int y =0;

s.o.p(y);

} // y dies here

s.o.p(y);

}

Int x =0;

s.o.p(x++); 0

s.o.p(++x); 1

int y =1;

s.o.p(y--); 1

s.o.p(--y); 0

for(int i=0;i<5;++i)

x+=1;

x=x+1;

x-=1;

x=x-1;

ARRAY:

Can hold the elements of same data type.

Give the size of the array while declaring.

Arrays work out w.r.t index.

Array’s index starts from 0.

But Array.length gives the appropriate size of the array.

Arr[5] = {6,7,8,9,10}

Int a[] = new int[5];

A[0] = 6;

A[1] = 7;

…

Int a[] = {10,20,30,40,50};

//Int a[][] = {{1,2}, {3,4}, {5,6}}

Constructors

By default any class will be supplied with 0 arg constructor.

Super()

Static, this, super, method overloading and method overriding, inheritance

Static – not an instance variable.

It is a class variable, global.

Variables, classes, blocks, methods -> can be static.

This operator:

It is a reference variable which points to the current objects.

Obj.name = “abc”;

This -> can be used on instance variables, passed as an argument, with the constructor also…

Super:

It is a reference variable which can point the super/parent class objects.

It can used with methods, constructors and variables.

Abstraction:

Hiding the implementation to the outside world.

Using the keyword abstract…

Abstract class -> there should be atleast one method which is abstract.

abstract void calculate();

void calculate();

interfaces:

100% abstract…

Class A extends B, C

Class A implements B, C

Interface Policy{

abstract Void calculate();

Public static final int a=1;

}

Exception handling:

Exception -> not as per plan…

String s;

s.o.p(s.length());

0

Null.length -> NPE -> Null Pointer Exception

Runtime errors..

String, File, Error, Exception etc…

Collection, Array

Throwable

Exception -> programmer should handle these…

Error

ArrayIndexOutOfBoundException

FileNotFoundException

ArithmeticException

NumberFormatException

Null Pointer Exception

ClassCastException

Premium/span

5/0;

ArthmeticException

Try, catch, Finally, throws, throw

Nested try -> yes possible

Multiple catch-> yes

Psvm(String args[]){

Import com.ust.test.\*;

Exception ->

Try{

Int x = 5/0;

}catch(Exception ex){

Ex.print();

}

}

File I/O

File is a class in Java.

Create a file

Delete a file

I can restrict access to the file

Copy of the file

File f = new File();

Java.io.\*

Byte

Byte data…

Read(), write()

Stream: flow of data/series of data

Byte stream and character stream

Buffer: temporary storage.

InputReader, OutputWriter

BufferedReader, Bufferedwriter

Collections in Java:

Collections is a class in java.

It can contain heterogeneous elements and can modify the size dynamically.

Collection is an interface.

List -> arraylist, Linkedlist ordered elements but can duplicates also!

Map -> HashMap (key, value)

Set -> TreeSet, HashSet (unique elements, no duplicates)

Queue -> PriorityQueue

ArrayList arr = new ArrayList();

arr.add();

arr.indexof()

arr.add(3);

arr.add(“abc”);

arr.add(12.34);

String s = arr.get(i);

Generic types came into picture…. <T>, <E>

ArrayList<String> strLst = new ArrayList<String>();

ArrayList<Employee> empLst = new ArrayList<Employee>();

Int, float, double -> data types

Wrapper classes:

Integer, Float, Double, Long, Character, Boolean -> Wrapper classes.

Integer i =new Integer(5);

(autoboxing)

int x = i.value();

(unboxing)

Iterator built in class for printing the values of any collection type.

for(int n:numbers)

Generics, Threads and Inner classes:

Thread:

Thread is a class which allows perform multitasking…

Using thread class

Runnable interface

Write my logic in a run() method

Class Example extends Thread

Class Example implements Runnable

{

Example e = new Example();

e.start();

}

e.isalive()

e.sleep()

public void run(){

}

Generics:

Number -> Integer-> Long-> Float->Double

ArrayList<T> arr = new ArrayList<T>();

Print(T[] elem){

for(E e: elem){

s.o.p(e);

}

Print(arr)

Number, Integer, Float

T

Number n = 5;

Integer i =6;

Float f = 7;

Vehicle -> Car, Bike

T -> Vehicle/Car/Bike

Inner classes:

class Outer{

class Inner{

}

}

Static inner class

Inner class created within a method

Anonymous Inner class

Regular inner class

Outer.Inner in = new Outer.Inner();

Testing Concepts:

Quality Center:

This is a tool:

WaterFall: The next phase starts after the earlier phase has fully completed.

Agile -> in the form of Sprints…each sprint spans upto around 2 weeks.

Scrum Master, Scrum Meeting / Standup meeting

The no. of requirements that can fit in a 2 weeks span.

The input for the Testing team is they have go through the User stories created by BA’s.

Requirement Tracebility Matrix(RTM): This is a table format which will have the mapping between the requirements and Testcases.

R1 TC1 link to the testcase doc -> XLS sheet.

RTM is been created by your Testing engineer/Test lead to have a track and mapping between all the requirements and the testcases – status.

Test Strategy: Test Approach – defines how the testing should be carried out!

Test plan: It is a detailed doc which describes the test strategy.

Schedule, estimates, Deliverables and test data etc - > Test plan.

Test Scenario: It is groups of testcases.

Given a requirement/usecase there can be many possibilities -> Test scenario.

When the test scenario is broken up - > Test case.

Test case-> emphasizes one particular flow/ has the steps to execute one single flow. This is written in a xls file.

Test Script: same as test case but created programmatically.

STLC:

Software Testing Life cycle.

Requirment is confirmed -> Test planning -> RTM -> Test Strategy -> Test plan -> Test Scenarios -> Test cases / Test scripts -> Test Execution -> Go / no Go (defect fixing by developers)

Bug and a defect:

Bug is a mistake in the program -> unit / dev-Integration phases.

Defect: this is a deviation in the requirement. The program is not behaving as per the requirement.

QA -> Quality Analyst-> part of the testing team.

Scope:

RTM:

Automation scope:

Unit testing

Dev/Integration testing

UAT – User Acceptance Testing

Pre – Production testing

Production testing

Smoke testing: If at all, all the basic functionalities are working fine.

Sanity testing: whenever a minor change is made, we check the entire application if its still working fine.

Functional testing: To see if all the functionalities working fine or not.

Performance testing: How fast the page is rendered. Not more then 3 mins (as specified by client).

Regression testing: Any new code has not impacted the existing functionality.

BlackBox testing: You don’t care on the internal code, you test the functionalities of the appln.

WhiteBox testing/open box testing/Code based testing/Glass box testing: The tester has the visibility on the flow of the data, structure of the code etc is available to the tester.

STLC phases:

Acceptance Criteria:

Expected behavior of the system as per the requirement.

Epic -> piece of user story

Entry and Exit criterias:

Test Management, Risk Analysis, QA/QE, Defects life cycle

Using a tool u need to manage the testcases and defects.

Defects life cycle:

New -> assign-> in progress-> Test-> re-test(to QA)->Closed / re-opened

QA / QE , test pyramid and V&V:

Quality assurance Team:

SQA

QA -> Testing role

QE: He puts in the stringent rules to what tools to be used and policing on the quality regulations so that the quality is assured.

Test Pyramid:

Some model or a structure that should be followed to have a quality product which is made faster and more efficient.

Integration/Component/Modules testing.

V&V:

Verification & Validation testing:

User stories.

FRD: Functional Requirement Document

BRD: Business Requirement Document

HLD: High level Design Document

LLD: Low Level Design Document

Requirement testing, types of it and Static testing

Requirement testing is based on the requirement provided by the client. All my test cases, test scenarios , test data are inclined from Requirements.

Functional(based on the req.) and non functional (performance, system hardware etc) testing:

Types of Requirement testing:

Implicit requirements:

Explicit requirement:

Latent requirements:

Static testing: (done without executing the appln.)

These dummy features:

1. Review:
2. Static analysis:

Alpha testing, beta testing

SQL:

Structured Query Language:

Datawarehouse

Database/Backend…

DATA

BIG DATA

HADOOP, SPARK, CLOUD Computing (Azure, AWS, GCP)

MYSQL : is a database system. It is a RDBMS – Relational Database Management system.

Structured: It is organized in the form of tables (row and column).

MYSQL is structured because its in the form of tables (row and column) and also has a schema.

Schema diagram / ER diagram (Entity relationship)

Schema is a blueprint.

Primary key / Foreign key definitions….

MongoDB is a NoSQL DB… -> it will store data in the form of documents…

No schema/No tables / No Rows and columns

Hbase: It is also a noSQL db, here it will store data in the form of hfiles..

DynamoDB it is also a noSQL DB…

SQL:

Create a table:

Insert data into the table

Select data from table

Delete unwanted data

CRUD operations: Create, Read, Update and Delete…

MySql Workbench

Port no: 3306

MySQL, Oracle -> RDBMS systems

Create table Employee(name varchar(30), address varchar(30), salary float, age int );

I need to create a database…

SQL data types:

Int, float, varchar(), Date, char, Blob

Create database dbName;

dbName;

use dbName;

create table Employee()

dbName1

create table Employee1()

Operators:

Arithmetic operators

Logical operators

Set var x = 3+5;

CRUD

Drop database dbName;

Drop table Employee;

Delete from Employee where name = “%Sir”;

Insert into table Employee values(“Emp1”, “XXXXXXXX”, 25.0000, 23);

JDBC, ODBC….

DriverManager

Connection

GroupBy

OrderBy

Joins

Write some complex queries

Select distinct

Where

Order by desc

Update Employee set mobile =”new value” where empId = “”;

10 – top

22 between 30

Like ‘%sir’

\*(zero or more occurences), %, [], -: WildCard characters

A\*

A

Aaaa

Ab

What is View? What is Stored Procedure?

What is indexing?

Table Patient(name) Table Doctor (name)

p.name, d.name

Patient name, age, disease, Doctor name, Prescription, Doctor Contact no

Select \* from Patient p, Doctor d where p.Did = d.id

Inner join

Outer / full join

Right outer join

Left outer join

Cross join

Inner join: return the matching records…

Left outer join: Return all the records from the left table + the matching records only from the right table

Right outer join: Returns all the records from the right table + only the matching records from the left table

Outer (Full) join:

This is a combined result of both left and right tables, but the records which don’t have a match for those fields NULL will be inserted.

Cross join: same as the cartesian product.

No. of rows in the first table \* No. of rows in the second table

SQL Queries, Joins, Subqueries, GroupBy, CRUD, NoSQL, MongoDB, CRUD , Aggregation

Views, Indexing

View: Virtual table….(it has rows and columns)

Create view vName as Select \* from Employee where age < 50;

Select \* from vName;

What is Index?

Indexes also to improve the performance…

w.r.t select statements

create index indexName on Employee (age, Salary)

Difference between Index and View?

Difference between Temporary table and View?

What is Stored procedure?

Stored Procedures are nothing but functions/Methods which will hold verified SQL statements…