**Notes: First you prepare how to write a code for syntax and write a code**

**Java: (day one)**

Steps to install java

Steps to install eclipse

Steps to create workspace

Steps to create project

File -> Project

We call Project is program

How to create .java file/class

Project -> new class and give extension .java

Class Employee {

}

how to create packages and what is best way to give name

From solution explorer, select project, right click and select package

Ex: companyname.projectname.foldername (this is common naming standard)

**what is main method will do?**

Main method is starting point of program

**What is variable?**

It will store the value in memory

To create variable we specify

Variablename datatype;

**what is data type and different data types`**

It will represents what type of data

Int

Double

Float

**creating property/data members** : we create properties at class level

int salary

String firstname

**creating method with void** : we write methods in

void

**creating method with void and parameter**

**package** vish;

**public** **class** voidparameter {

**public** **static** **void** main(String[] args) {

//System.out.println("The sum of the integers is " +sum);

vinod v=**new** vinod();

v.vinod();

}

**public** **static** **class** vinod{

**int** a =1;

**int** b=9;

**int** sum=(a+b);

**public** **void** vinod(){

System.***out***.println("The sum of the integers is " +sum);

}

}

}

**creating method with return data type**

**package** vish;

**public** **class** returndata {

**public** **static** **void** main(String[] args)

{

*vinod*(77,50);

*vinod*(20,30);

}

**public** **static** **int** vinod (**int** a,**int** b)

{

System.***out***.println(a+b);

**return** (a+b);

}

}

**creating method with return data type and parameter**

**package** vish;

**public** **class** returndata {

**public** **static** **void** main(String[] args)

{

*vinod*(77,50);

*vinod*(20,30);

}

**public** **static** **int** vinod (**int** a,**int** b)

{

System.***out***.println(a+b);

**return** (a+b);

}

}

**creating variable**

**creating static property All instances shared the value http://crunchify.com/java-static-methods-variables-static-block-and-class-with-example/**

**creating static method**

**http://crunchify.com/java-static-methods-variables-static-block-and-class-with-example/**

**creating object**

**calling method with no return**

**package** vish;

**public** **class** returndata {

**public** **static** **void** main(String[] args)

{

*vinod*(771,50);

*vinod*(20,30);

}

**public** **static** **void** vinod (**int** a,**int** b)

{

System.***out***.println(a+b);

}

}

**calling method with no return and parameter**

**package** vish;

**public** **class** returndata {

**public** **static** **void** main(String[] args)

{

*vinod*(771,50);

*vinod*(20,30);

}

**public** **static** **void** vinod (**int** a,**int** b)

{

System.***out***.println(a+b);

}

}

**calling method with return and no parameter**

**package** vish;

**public** **class** data1 {

**public** **static** **void** main(String[] args) {

vinod vish=**new** vinod();

vish.data1();

//System.out.println(c);

}

**public** **static** **class** vinod{

**public** **int** data1(){

**int** a=2;

**int** b=20;

**int** c=a+b;

System.***out***.println(c);

**return** (c);

}

}

}

**calling method with return and parameter**

**package** vish;

**public** **class** data1 {

**public** **static** **void** main(String[] args) {

// **TODO** Auto-generated method stub

**int** a=100+100;

System.***out***.println(a);

}

**public** **static** **int** data1(**int** a){

System.***out***.println(a);

**return** (a);

}

}

**calling method with return and storing the return data**

**package** vish;

**public** **class** data1 {

**public** **static** **void** main(String[] args)

{

*vinod*("vinodkumar");

*vinod*("welcomes you");

**int** sum=*data1*(50,90);

System.***out***.println(sum);

**int** sum1=*data1*(500,10);

System.***out***.println(sum1);

}

**public** **static** **void** vinod(String name)

{

System.***out***.println("Hello " + name);

}

**public** **static** **int** data1 (**int** a,**int** b)

{

**return** (a+b);

}

}

**calling static method**

**package** vinod;

**public** **class** Calstatic {

**static** **int** *x*=10;

**void** nonstaticMethod(){

*x*=200;

System.***out***.println("X=" +*x*);

*staticMethod*();

}

**public** **static** **void** staticMethod(){

*x*=409;

}

**public** **static** **void** main(String[] args) {

Calstatic obj=**new** Calstatic();

System.***out***.println("X="+ *x*);

obj.nonstaticMethod();

System.***out***.print("X=" +*x*);

}

}

**using static property**

**Java: Day2**

**create classes under multiple packages**

**calling classes under different packages**

**write code to handle exceptions with try/catch/finally**

**what is final keyword**

**It can’t sub classed, overridden and only initialized once**

**package Lesson1;**

**public class Keyword {**

**public static void main(String[] args) {**

**hell hel=new hell();**

**hel.number=10;**

**//System.out.println(hel);**

**}**

**}**

**package Lesson1;**

**public class hell {**

**public static int number;**

**hell(){**

**int number=0;**

**while(number<=15){**

**System.out.println(number);**

**number++;**

**}**

**}**

**}**

**write code for interface and create class to implement that interface**

**write code for creating abstract class (use polymorphism)(Concrete is vice versa)**

**It can’t have instantiated, not make object and it implemented by subclasses**

**1.if u make method abstract and need to give implementation in subclass**

**package** Lesson1;

**public** **class** overriding {

**public** **static** **void** main(String[] args) {

Bank abc=**new** Bank\_ABC();

System.out.println(abc.getInterestRate());

}

}

**package Lesson1;**

**abstract public class Bank {**

**abstract int getInterestRate(){**

**return 0 ;**

**}**

**}**

**package Lesson1;**

**public class Bank\_ABC extends Bank{**

**int getInterestRate(){**

**return 9;**

**}**

**}**

**implement method overloading**

**package** vish;

**public** **class** BoABank {

**public** **static** **void** main(String[] args) {

System.***out***.println(*Add*(133333333,3));

System.***out***.println(*Add*(1.2,3.2));

System.***out***.println(*Add*("Vinod"," padidala"));

}

**public** **static** **int** Add (**int** a,**int** b){

**return** (a+b);

}

**public** **static** Double Add (Double a,Double b){

**return**(a+b);

}

**public** **static** String Add (String a,String b){

**return**(a+b);

}

}

**implement method overriding**

**package** Lesson1;

**public** **class** overriding {

**public** **static** **void** main(String[] args) {

Bank abc=**new** Bank\_ABC();

System.out.println(abc.getInterestRate());

}

}

**package Lesson1;**

**public class Bank {**

**int getInterestRate(){**

**return 0 ;**

**}**

**}**

**package Lesson1;**

**public class Bank\_ABC extends Bank{**

**int getInterestRate(){**

**return 9;**

**}**

**}**

**implementing polymorphism**

**package** Lesson1;

**public** **class** polymorphism {

**public** **static** **void** main(String[] args) {

Bank abc=**new** Bank\_ABC();

Bank def=**new** Bank\_DEF();

Bank xyz=**new** Bank\_XYZ();

Bank ghi=**new** Bank\_GHI();

System.out.println(abc.getInterestRate());

System.out.println(def.getInterestRate());

System.out.println(xyz.getInterestRate());

System.out.println(ghi.getInterestRate());

}

}

**package Lesson1;**

**public class Bank\_ABC extends Bank{**

**int getInterestRate(){**

**return 9;**

**}**

**}**

**package Lesson1;**

**public class Bank\_DEF extends Bank{**

**int getInterestRate(){**

**return 12;**

**}**

**}**

**package Lesson1;**

**public class Bank\_GHI extends Bank{**

**int getInterestRate(){**

**return 6;**

**}**

**}**

**package Lesson1;**

**public class Bank\_XYZ extends Bank{**

**int getInterestRate(){**

**return 22;**

**}**

**}**

**package Lesson1;**

**public class Bank {**

**int getInterestRate(){**

**return 0 ;**

**}**

**}**

**implementing interface**

Main **use of Interface** is that it overcome the problems of Multiple Inheritance.

**It can’t have instantiated and all the subclass/members of interface is abstracted by nature**

**It can implementes not extend**

**Class class—extends**

**Interface class-----imolements**

**Interface interface---extends**

**package** vish;

**public** **class** BoABank {

**public** **static** **void** main(String[] args) {

}

}

**package** vinod;

**public** **interface** bank {

**int** getinterestrate();

}

**package** vinod;

**public** **class** bank\_abc **implements** bank{

**public** **int** getinterestrate(){

**return** 6;

}

}

**write a code to save data into excel file and read from excel file (POI and jexcel API)**

**write a code to save data into excel file**

**package** vinod;

**import** java.io.File;

**import** java.io.FileOutputStream;

**import** java.util.Map;

**import** java.util.Set;

**import** java.util.TreeMap;

**import** org.apache.poi.ss.usermodel.Cell;

**import** org.apache.poi.ss.usermodel.Row;

**import** org.apache.poi.xssf.usermodel.XSSFSheet;

**import** org.apache.poi.xssf.usermodel.XSSFWorkbook;

**public** **class** excel {

**public** **static** **void** main(String[] args) {

//Blank workbook

XSSFWorkbook workbook = **new** XSSFWorkbook();

//Create a blank sheet

XSSFSheet sheet = workbook.createSheet("Employee Data");

//This data needs to be written (Object[])

Map<String, Object[]> data = **new** TreeMap<String, Object[]>();

data.put("1", **new** Object[] {"ID", "NAME", "LASTNAME"});

data.put("2", **new** Object[] {1, "Amit", "Shukla"});

data.put("3", **new** Object[] {2, "Lokesh", "Gupta"});

data.put("4", **new** Object[] {3, "John", "Adwards"});

data.put("5", **new** Object[] {4, "Brian", "Schultz"});

//Iterate over data and write to sheet

Set<String> keyset = data.keySet();

**int** rownum = 0;

**for** (String key : keyset)

{

Row row = sheet.createRow(rownum++);

Object [] objArr = data.get(key);

**int** cellnum = 0;

**for** (Object obj : objArr)

{

Cell cell = row.createCell(cellnum++);

**if**(obj **instanceof** String)

cell.setCellValue((String)obj);

**else** **if**(obj **instanceof** Integer)

cell.setCellValue((Integer)obj);

}

}

**try**

{

//Write the workbook in file system

FileOutputStream out = **new** FileOutputStream(**new** File("how to do in java\_demo.xlsx"));

workbook.write(out);

out.close();

System.***out***.println("how to do in java\_demo.xlsx written successfully on disk.");

}

**catch** (Exception e)

{

e.printStackTrace();

}

}

}

**write code and read from excel file**

**how to update the data into XML file and read data from XML file**

**write code to add items to integer, string array**

**write code to retrieve items from integer, string array**

**write code to add items to ArrayList collection**

**write code to retrieve items from arraylist (using for each loop\_**

**write code to add items HashMap**

**write code to retrieve items HashMap**

**Write code to add items to hashset and Write code to retrieve items to hasset**

import java.util.Collections;

import java.util.Enumeration;

import java.util.HashSet;

import java.util.Iterator;

import java.util.Set;

public class SetDemo

{

public static void main(String args[])

{

Set set = new HashSet();

//Adding values to the HashSet

set.add("test1");

set.add("test2");

set.add("test3");

System.out.println("Retrieving values from HashSet using Iterator");

retrieveValuesFromListMethod1(set);

System.out.println("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n\n");

System.out.println("Retrieving values from HashSet using Enumeration");

retrieveValuesFromListMethod2(set);

System.out.println("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n\n");

}

/\*This method retrieves values from HashSet using Iterator

\*/

public static void retrieveValuesFromListMethod1(Set set)

{

Iterator itr = set.iterator();

while(itr.hasNext())

{

System.out.println(itr.next());

}

}

/\*This method retrieves values from HashSet using Enumeration

\*/

public static void retrieveValuesFromListMethod2(Set set)

{

Enumeration e = Collections.enumeration(set);

while(e.hasMoreElements())

{

System.out.println(e.nextElement());

}

}

}

**write code to connect to JDBC to get rows from employee table**

**package vinod;**

**import java.sql.Connection;**

**import java.sql.DriverManager;**

**import java.sql.ResultSet;**

**import java.sql.SQLException;**

**import java.sql.Statement;**

**import org.testng.annotations.Test;**

**public class ConnectMySQL {**

**@Test**

**public void testDB() throws ClassNotFoundException, SQLException{**

**//DriverManager.registerDriver(new oracle.jdbc.OracleDriver());**

**//FileOutputStream fos=new FileOutputStream("F:\\data.doc");**

**Class.forName("com.mysql.jdbc.Driver");**

**System.out.println("Driver loaded \n");**

**Connection con=DriverManager.getConnection("jdbc:mysql://127.0.0.1:3306/vinod\_DB","root","root");**

**System.out.println("Connected to MySQL DB\n");**

**Statement smt=con.createStatement();**

**ResultSet rs=smt.executeQuery("select \* from vinod\_table");**

**while(rs.next())**

**{**

**String firstname=rs.getString("name");**

**System.out.println("Database name "+firstname);**

**String Email=rs.getString("mail");**

**System.out.println("Database mail "+Email);**

**}**

**}**

**}**

**create Employee class**

**Add employee class to list collection**

**create method that return list of employee collection**

**Difference between string, string buffer, string builder with example**