**Java Day: 2**

create classes under multiple packages

calling classes under different packages

18. write code to handle exceptions with try/catch/finally

public class Exmpl {

public static void main(String args[]) {

int a[] = new int[4];

try {

System.out.println("Access element three :" + a[5]);

}catch(ArrayIndexOutOfBoundsException e) {

System.out.println("Exception thrown :" + e);

}finally {

a[0] = 10;

System.out.println("First element value: " + a[0]);

System.out.println("The finally statement is executed");

}

}

}

19. what is final keyword

Ans: The **final keyword** in java is used to restrict the user. The java final keyword can be used in many context. Final can be:

1. variable
2. method
3. class

The final keyword can be applied with the variables, a final variable that have no value it is called blank final variable or uninitialized final variable. It can be initialized in the constructor only. The blank final variable can be static also which will be initialized in the static block only

20. write code for interface and create class to implement that interface

Ans :

interface Employee {

public void eno( );

public void salary( );

}

**Implementation:**

interface MyInterface

{ public void method1();

public void method2();

}

Class Example implements MyInterface

{

public void method1()

{ System.out.println(“implementation of method1”); }

public void method2()

{

System.out.println(“implementation of method2”);

}

public static void main(String arg[])

{ Myinterface obj = new Example();

obj.method1(); }

21. write code for creating abstract class

**Ans:**

**abstract** **class** Bike{

**abstract** **void** run();

}

**class** Honda **extends** Bike{

**void** run(){ System.out.println("running safely.."); }

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

 Bike obj = **new** Honda();

obj.run();

}

}

22. implement method overloading

Ans : class Display

{

public

Ans :

class DisplayOverloading

{

public void disp(char c)

{

System.out.println(c);

}

public void disp(char c, int num)

{

System.out.println(c + " "+num);

}

}

class Sample

{

public static void main(String args[])

{

DisplayOverloading obj = new DisplayOverloading();

obj.disp('a');

obj.disp('a',10);

}

}

implement method overriding

implementing polymorphism

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

**Read from excel:**

public static void main(String[] args) throws IOException {

        String excelFilePath = "Books.xlsx";

        FileInputStream inputStream = new FileInputStream(new File(excelFilePath));

        XSSFWorkbook workbook = new XSSFWorkbook(inputStream);

        XSSFSheet firstSheet = workbook.getSheetAt(0);

        Iterator<Row> iterator = firstSheet.iterator();

        while (iterator.hasNext()) {

            Row nextRow = iterator.next();

            Iterator<Cell> cellIterator = nextRow.cellIterator();

            while (cellIterator.hasNext()) {

                Cell cell = cellIterator.next();

                switch (cell.getCellType()) {

                    case Cell.CELL\_TYPE\_STRING:

                        System.out.print(cell.getStringCellValue());

                        break;

                    case Cell.CELL\_TYPE\_BOOLEAN:

                        System.out.print(cell.getBooleanCellValue());

                        break;

                    case Cell.CELL\_TYPE\_NUMERIC:

                        System.out.print(cell.getNumericCellValue());

                        break;

                }

                System.out.print(" - ");

            }

            System.out.println();

        }

        workbook.close();

        inputStream.close();

    }

**Write into Excel :**

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

XSSFSheet sheet = workbook.createSheet("Sample sheet");

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

data.put("1", **new** Object[] {"empNo.", "name", "salary"});

data.put("2", **new** Object[] {1, "John", 1500000d});

data.put("3", **new** Object[] {2, "Sam", 800000d});

data.put("4", **new** Object[] {3, "Dean", 700000d});

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** Integer)

cell.setCellValue((Integer)obj);

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

cell.setCellValue((String)obj);

**else** **if**(obj **instanceof** Double)

cell.setCellValue((Double)obj);

}

}

**try** {

//new excel file created by fileoutput stream object

FileOutputStream out =

**new** FileOutputStream(**new** File("C:\\Users\\Public\\Nagarjuna2.xlsx"));

workbook.write(out);

out.close();

System.***out***.println("Excel written successfully..");

} **catch** (FileNotFoundException e) {

e.printStackTrace();

} **catch** (IOException e) {

e.printStackTrace();

}

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

**Read XML file data:**

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

**try** {

File fXmlFile = **new** File("C:\\Users\\Arjun\\Desktop\\Sample.xml");

DocumentBuilderFactory dbFactory = DocumentBuilderFactory.*newInstance*();

DocumentBuilder dBuilder = dbFactory.newDocumentBuilder();

Document doc = dBuilder.parse(fXmlFile);

//optional, but recommended

//read this - http://stackoverflow.com/questions/13786607/normalization-in-dom-parsing-with-java-how-does-it-work

doc.getDocumentElement().normalize();

System.***out***.println("Root element :" + doc.getDocumentElement().getNodeName());

NodeList nList = doc.getElementsByTagName("staff");

System.***out***.println("----------------------------");

**for** (**int** temp = 0; temp < nList.getLength(); temp++) {

Node nNode = nList.item(temp);

System.***out***.println("\nCurrent Element :" + nNode.getNodeName());

**if** (nNode.getNodeType() == Node.***ELEMENT\_NODE***) {

Element eElement = (Element) nNode;

System.***out***.println("Staff id : " + eElement.getAttribute("id"));

System.***out***.println("First Name : " + eElement.getElementsByTagName("firstname").item(0).getTextContent());

System.***out***.println("Last Name : " + eElement.getElementsByTagName("lastname").item(0).getTextContent());

System.***out***.println("Nick Name : " + eElement.getElementsByTagName("nickname").item(0).getTextContent());

System.***out***.println("Salary : " + eElement.getElementsByTagName("salary").item(0).getTextContent());

}

}

} **catch** (Exception e) {

e.printStackTrace();

}

}