**Java Lab Program Questions**

**PART -A**

1. **Design a java Program to add two matrix using multi-dimensional arrays, pass two dimensional array as parameter to the method**

**import** java.util.Scanner;

**public** **class** Msum {

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

{

**int** i, j;

**int** mat1[][] = **new** **int**[3][3];

**int** mat2[][] = **new** **int**[3][3];

**int** mat3[][] = **new** **int**[3][3];

Scanner sc = **new** Scanner(System.***in***);

System.***out***.println("Enter Matrix 1 Elements : ");

**for**(i=0; i<3; i++)

{

**for**(j=0; j<3; j++)

{

mat1[i][j] = sc.nextInt();

}

}

System.***out***.println("Enter Matrix 2 Elements : ");

**for**(i=0; i<3; i++)

{

**for**(j=0; j<3; j++)

{

mat2[i][j] = sc.nextInt();

}

}

**for**(i=0; i<3; i++)

{

**for**(j=0; j<3; j++)

{

mat3[i][j] = mat1[i][j] + mat2[i][j];

}

}

System.***out***.println("Sum of matrices:");

**for**(i=0; i<3; i++)

{

**for**(j=0; j<3; j++)

{

System.***out***.print(mat3[i][j]+ " ");

}

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

}

}

}

**OR**

**import** java.util.\*;

**class** sum{

**void** dosum(**int** [][]a,**int** [][]b,**int** m,**int** n) {

**int** i,j;

**int** [][]sum=**new** **int**[m][n];

System.***out***.println("resulting matrix sum:");

**for**(i=0;i<m;i++) {

**for**(j=0;j<n;j++) {

sum[i][j]=a[i][j]+b[i][j];

}

}

**for**(i=0;i<m;i++) {

**for**(j=0;j<n;j++) {

System.***out***.print(sum[i][j]+"\t");

}

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

}

}

}

**public** **class** matrixmul {

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

**int** m,n;

System.***out***.println("enter the no of rows and columns");

Scanner obj=**new** Scanner(System.***in***);

m=obj.nextInt();

n=obj.nextInt();

**int** [][]a=**new** **int**[m][n];

**int** [][]b=**new** **int**[m][n];

**int** i,j;

System.***out***.println("enter the elements of MATRIX a:");

**for**(i=0;i<m;i++) {

**for**(j=0;j<n;j++) {

a[i][j]=obj.nextInt();

}

}

System.***out***.println("enter the elements of MATRIX b:");

**for**(i=0;i<m;i++) {

**for**(j=0;j<n;j++) {

b[i][j]=obj.nextInt();

}

}

sum o=**new** sum();

o.dosum(a,b,m,n);

}

}

1. **Design a java program with one method to put even & odd elements of an array in 2 separate arrays, and another method to find the transpose of the matrix by implementing the concept of method overriding.**

package hybrid;

import java.util.\*;

class evenodd{

void func() {

int m,n,i,j;

System.out.println("enter the no of rows and columns");

Scanner obj=new Scanner(System.in);

m=obj.nextInt();

n=obj.nextInt();

int [][]a=new int[m][n];

int [][]b=new int[m][n];

System.out.println("enter the elements of MATRIX a:");

for(i=0;i<m;i++) {

for(j=0;j<n;j++) {

a[i][j]=obj.nextInt();

}

}

System.out.println("Transpose of matrix a:");

for(i=0;i<m;i++) {

for(j=0;j<n;j++) {

b[j][i]=a[i][j];

}

}

for(i=0;i<m;i++) {

for(j=0;j<n;j++) {

System.out.print(b[i][j]+"\t");

}

System.out.println();

}

}

}

class transpose extends evenodd{

void func() {

int m,i,k=0,l=0;

System.out.println("enter the size of array");

Scanner obj=new Scanner(System.in);

m=obj.nextInt();

int []a=new int[m];

int []even=new int[m];

int []odd=new int[m];

System.out.println("enter the array elemnets:");

for(i=0;i<m;i++) {

a[i]=obj.nextInt();

}

for(i=0;i<m;i++) {

if(a[i]%2==0) {

even[k]=a[i];

k++;

}

else {

odd[l]=a[i];

l++;

}

}

System.out.println("even array elements:");

for(i=0;i<k;i++) {

System.out.print(even[i]+"\t");

}

System.out.println();

System.out.println("odd array elements:");

for(i=0;i<l;i++) {

System.out.println(odd[i]+"\t");

}

}

}

public class prg2{

public static void main(String[] args) {

evenodd a=new evenodd();

evenodd b=new transpose();

a.func();

b.func();

}

}

1. **Design a java code to accept a value for ‘n’ and calculate the total number of all possible squares in a square matrix.**

**import** java.util.Scanner;

**public** **class** SqareMatrix

{

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

{

Scanner in=**new** Scanner(System.***in***);

**while**(**true**)

{

System.***out***.print("Enter the value of n for n-Square Matrix : ");

**int** n=in.nextInt();

**if**(n<=0)

System.***out***.println("invalid input");

**else**

{

**int** sum=0;

**while**(n!=0)

{

sum=sum+n\*n;

n--;

}

System.***out***.println("The number square matrices are : " +sum);

}

}

}

}

1. **Design a code for a simple calculator which takes input from the user and also details of what operation must be performed. The user can input only 2 operands**

**import** java.util.Scanner;

**public** **class** TwoOperands

{

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

{

Scanner in=**new** Scanner(System.***in***);

**int** ch=0;

**do**

{

System.***out***.println("Please enter value for first operand : ");

**double** a=in.nextDouble();

System.***out***.println("Please enter value for second operand : ");

**double** b=in.nextDouble();

System.***out***.println("Select the operation to be performed");

System.***out***.println("1. Addition");

System.***out***.println("2. Subtraction");

System.***out***.println("3. Multiplication");

System.***out***.println("4. Division");

System.***out***.println("5. Exit");

System.***out***.print("Enter the option : ");

ch=in.nextInt();

**switch**(ch)

{

**case** 1:

System.***out***.println("The addition of two numbers is : " +(a+b));

**break**;

**case** 2:

System.***out***.println("The subtraction of two numbers is : " +(a-b));

**break**;

**case** 3:

System.***out***.println("The multiplication of two numbers is : " +(a\*b));

**break**;

**case** 4:

System.***out***.println("The division of two numbers is : " +(a/b));

**break**;

**case** 5:

System.*exit*(0);

**default**:

System.***out***.println("Please enter valid option");

**break**;

}

}

**while**(ch!=5);

}

}

## Design a java program to calculate the difference between the sum of the odd level and even level nodes of a binary tree.

import java.util.ArrayList;

import java.util.LinkedList;

import java.util.Queue;

class GFG{

// Structure of a node

// of an n-ary tree

static class Node

{

    int val;

    ArrayList<Node> children;

    public Node(int val)

    {

        this.val = val;

        this.children = new ArrayList<Node>();

    }

};

static class Pair

{

    Node first;

    int second;

    public Pair(Node node, int val)

    {

        this.first = node;

        this.second = val;

    }

}

// Function to find the difference

// between of sums node values of

// odd and even levels in an N-ary tree

static int evenOddLevelDifference(Node root)

{

    // Store the sums of nodes at

    // even and odd levels

    int evenSum = 0, oddSum = 0;

    // Initialize a queue to store

    // pair of node and level

    Queue<Pair> q = new LinkedList<>();

    // Push the root into the

    // queue with level 1

    q.add(new Pair(root, 1));

    // Iterate all levels

    // of tree are traversed

    while (!q.isEmpty())

    {

        // Store the node at the

        // front of the queue

        Pair currNode = q.poll();

        // Store the current level

        int currLevel = currNode.second;

        // Store the current node value

        int currVal = currNode.first.val;

        // If current node

        // level is odd

        if (currLevel % 2 == 1)

            // Add to odd sum

            oddSum += currVal;

        else

            // Add to even sum

            evenSum += currVal;

        // Push all the children of current node

        // with increasing current level by 1

        for(Node child : currNode.first.children)

        {

            q.add(new Pair(child, currLevel + 1));

        }

    }

    // Return the difference

    return (oddSum - evenSum);

}

// Driver Code

public static void main(String[] args)

{

    // Create the N-ary Tree

    Node root = new Node(4);

    root.children.add(new Node(2));

    root.children.add(new Node(3));

    root.children.add(new Node(-5));

    root.children.get(0).children.add(new Node(-1));

    root.children.get(0).children.add(new Node(3));

    root.children.get(2).children.add(new Node(-2));

    root.children.get(2).children.add(new Node(6));

    System.out.println(evenOddLevelDifference(root));

}

}

OR

import java.util.\*;

import java.io.\*;

class Node{

int data;

Node left,right;

Node(int key){

data=key;

left=right=null;

}

}

class evenodddiff{

int difference(Node root) {

if(root==null)

return 0;

int level=0;

int evensum=0;

int oddsum=0;

Queue<Node> q=new LinkedList<>();

q.add(root);

while(q.size()!=0) {

int size=q.size();

level++;

while(size>0) {

Node temp=q.remove();

if(level%2==0) {

evensum+=temp.data;

}

else {

oddsum+=temp.data;

}

if(temp.left!=null) {

q.add(temp.left);

}

if(temp.right!=null) {

q.add(temp.right);

}

size--;

}

}

return oddsum-evensum;

}

}

class prg2{

public static void main(String []args) {

Node root=new Node(5);

root.left=new Node(2);

root.right=new Node(1);

root.right.left=new Node(3);

root.right.right=new Node(6);

root.right.left.left=new Node(4);

evenodddiff o=new evenodddiff();

int result=o.difference(root);

System.out.println("difference between sum of even level and odd level elements is:"+result);

}

}

1. **Design a java code to accept 5 strings from the user and print them in lexicographical order.**

**import** java.util.Arrays;

**import** java.util.Scanner;

**public** **class** Lexicographical

{

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

{

Scanner in=**new** Scanner(System.***in***);

String[] ls=**new** String[5];

System.***out***.println("Enter five strings : ");

System.***out***.print("First : ");

ls[0]=in.next();

System.***out***.print("Second : ");

ls[1]=in.next();

System.***out***.print("Third : ");

ls[2]=in.next();

System.***out***.print("Fourth : ");

ls[3]=in.next();

System.***out***.print("Fifth : ");

ls[4]=in.next();

Arrays.*sort*(ls);

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

{

System.***out***.println(ls[i]);

}

}

}

**OR**

**import** java.util.\*;

**public** **class** Sortstrings {

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

{

Scanner sc = **new** Scanner(System.***in***);

String []words= **new** String[5];

System.***out***.println("Enter names");

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

words[i] = sc.nextLine();

}

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

**for** (**int** j = 0; j < 5-i-1;j++){

**if** (words[i].compareTo(words[j+1]) > 0)

{

String temp=words[j];

words[j]=words[j+1];

words[j+1]=temp;

}

}

}

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

{

System.***out***.print(words[i]+ " ");

}

}

}

1. **Design a java code for implementing Binary Search, pass array as parameter to the method.**

**class** BinarySearchExample{

**public** **static** **void** binarySearch(**int** arr[], **int** first, **int** last, **int** key){

**int** mid = (first + last)/2;

**while**( first <= last ){

**if** ( arr[mid] < key ){

        first = mid + 1;

      }**else** **if** ( arr[mid] == key ){

        System.out.println("Element is found at index: " + mid);

**break**;

      }**else**{

         last = mid - 1;

      }

      mid = (first + last)/2;

   }

**if** ( first > last ){

      System.out.println("Element is not found!");

   }

 }

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

**int** arr[] = {10,20,30,40,50};

**int** key = 30;

**int** last=arr.length-1;

        binarySearch(arr,0,last,key);

 }

}

1. **Design a java code to implement method overloading.**
2. **class** Adder{
3. **static** **int** add(**int** a,**int** b){**return** a+b;}
4. **static** **int** add(**int** a,**int** b,**int** c){**return** a+b+c;}
5. }
6. **class** TestOverloading1{
7. **public** **static** **void** main(String[] args){
8. System.out.println(Adder.add(11,11));
9. System.out.println(Adder.add(11,11,11));
10. }}

**OR**

1. **class** Adder{
2. **static** **int** add(**int** a, **int** b){**return** a+b;}
3. **static** **double** add(**double** a, **double** b){**return** a+b;}
4. }
5. **class** TestOverloading2{
6. **public** **static** **void** main(String[] args){
7. System.out.println(Adder.add(11,11));
8. System.out.println(Adder.add(12.3,12.6));
9. }}

OR

import java.util.\*;

import java.io.\*;

class operations{

void printing(int x) {

System.out.println("the value is"+x);

}

void equality(int a,int b) {

System.out.println("checking whether the values of a and b are equal:");

if(a==b)

System.out.println("answer is :"+true);

else

System.out.println("answer is :"+false);

}

}

class prg2{

public static void main(String []args) {

int a,b;

Scanner s=new Scanner(System.in);

System.out.println("enter the elements a and b:");

a=s.nextInt();

b=s.nextInt();

operations obj=new operations();

obj.printing(a);

obj.printing(b);

obj.equality(a,b);

}

}

1. **Design a java program to implement multiple inheritance using an interface.**

interface InterfaceOne{

public void display();

}

interface InterfaceTwo{

public void display();

}

class Main implements InterfaceOne,InterfaceTwo{

public void display(){

System.out.println("Display() method implementation");

}

public static void main(String args[]){

Main obj = new Main();

obj.display();

}

}

**OR**

import java.util.\*;

import java.io.\*;

interface one{

void sum(int a,int b);

}

interface two{

void difference(int a,int b);

}

class ABC implements one,two{

public void sum(int a,int b) {

int sum=a+b;

System.out.println("sum of"+a+"and"+b+"is:"+sum);

}

public void difference(int a,int b) {

int difference=a-b;

System.out.println("Difference of"+a+"and"+b+"is:"+difference);

}

}

class prg2{

public static void main(String []args) {

ABC obj=new ABC();

Scanner s=new Scanner(System.in);

int a,b;

System.out.println("enter the values of a and b:");

a=s.nextInt();

b=s.nextInt();

obj.sum(a,b);

obj.difference(a,b);

}

}

1. **Design a code to print a pyramid based on level entered by the user**

**Enter the number of levels for the number pyramid : 6**

**1**

**2 3 2**

**3 4 5 4 3**

**4 5 6 7 6 5 4**

**5 6 7 8 9 8 7 6 5**

**6 7 8 9 10 11 10 9 8 7 6**

**import** java.util.Scanner;

**public** **class** Pyramid

{

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

{

Scanner in=**new** Scanner(System.***in***);

System.***out***.print("Enter the number of levels for the number pyramid : ");

**int** levels=in.nextInt();

**for**(**int** row=1;row<levels+1;row++)

{

**for**(**int** space=row;space<levels;space++)

{

System.***out***.print(" ");

}

**for**(**int** col=row;col<=2\*row-1;col++)

{

System.***out***.print(col+" ");

}

**for**(**int** k=2\*row-2;k>=row;k--)

{

System.***out***.print(k+" ");

}

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

}

}

}

1. **Write a Java program to find digit sum two digit number until the digit sum is less than 10.**

**import** java.util.Scanner;

**public** **class** DigitSum

{

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

{

Scanner in=**new** Scanner(System.***in***);

System.***out***.print("Enter the number to find digitsum : ");

**int** num=in.nextInt();

**int** sum=num;

**int** fin=0;

**int** fin1=0;

String temp=Integer.*toString*(num);

**int** len=temp.length();

**for**(**int** i=0;i<len;i++)

{

System.***out***.println("1 : " +sum%10);

fin=fin+sum%10;

**if**(Integer.*toString*(fin).length()!=1)

{

**for**(**int** j=0;j<=Integer.*toString*(fin).length();j++)

{

System.***out***.println("2 : " +fin%10);

fin1=fin1+fin%10;

fin=fin/10;

}

fin=fin1;

}

sum=sum/10;

}

System.***out***.println("3 : " +fin);

}

}

**OR**

**import** java.util.\*;

**public** **class** Digsum {

**public** **static** **void** calc(**int** num)

{

**int** sum = 0;

**while**(num > 0 || sum > 9)

{

**if**(num == 0)

{

num = sum;

sum = 0;

}

sum += num % 10;

num /= 10;

}

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

}

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

Scanner sc = **new** Scanner(System.***in***);

System.***out***.print("Enter number:");

**int** n=sc.nextInt();

*calc*(n);

}

}

1. **Design a java code which accepts a number (which is non-zero and positive) from the user and then checks if it is a happy number or not. Implement it using nested interface concept**

package hybrid;

import java.util.\*;

import java.io.\*;

interface happy{

interface number{

void CheckHappy(int n);

}

}

class Test implements happy.number{

public void CheckHappy(int n) {

int x,l,k,want;

int key=n;

do {

want=0;

while(n!=0) {

x=n/10;

l=n%10;

want+=l\*l;

n=x;

}

System.out.println(want);

n=want;

}while(want>=10);

if(want==1)

System.out.println("the number"+key+"is happy number");

else

System.out.println("the number is not happy number");

}

}

class prg2{

public static void main(String []args) {

Scanner o=new Scanner(System.in);

System.out.println("enter the number:");

int n=o.nextInt();

happy.number obj=new Test();

obj.CheckHappy(n);

}

}

**OR**

**import** java.util.\*;

**interface** yes{

**int** checkHappyNumber(**int** number);

}

**class** Happy **implements** yes {

**public** **int** checkHappyNumber(**int** number)

{

**int** rem = 0, sum = 0;

**while**(number > 0)

{

rem = number %10;

sum = sum+(rem\*rem);

number = number/10;

}

**return** sum;

}

}

**public** **class** example{

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

{

Happy ob=**new** Happy();

Scanner sc = **new** Scanner (System.***in***);

System.***out***.print("Enter a non-zero Positive Number:");

**int** number = sc.nextInt( );

**int** result = number;

**while** (result != 1 && result != 4)

{

result = ob.checkHappyNumber(result);

}

**if** (result ==1)

{

System.***out***.println ("It is a Happy Number");

}

**else**

{

System.***out***.println (" It is not a Happy Number");

}

}

}

1. **Design an java code which accepts a Room Number, Mobile Number and Name of the Customer and generate a 6 Character Unique Password**

**import** java.util.Scanner;

**public** **class** UniquePassword

{

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

{

Scanner s=**new** Scanner(System.***in***);

String name;

String pw=**new** String();

**long** ph;**int** rno,rem,sum=0;

System.***out***.println("Enter the name : ");

name=s.next();

System.***out***.println("Enter the room no : ");

rno=s.nextInt();

**int** rno1=rno;

System.***out***.println("Enter the phone number : ");

ph=s.nextLong();

**char** []sym={')','!','@','#','$','%','^','&','\*','('};

**do**

{

**while**(rno!=0)

{

rem=rno%10;

sum=sum+rem;

rno=rno/10;

}

**if**(sum>=10)

{

rno=sum;

sum=0;

}

}**while**(rno!=0);

**int** c=rno1/100;

**int** len=name.length();

**int** pos=0;

**for**(**int** i=0;i<len;i++)

{

**if**(c==i)

{

pos=i;

System.***out***.println("Password is : ");

}

}

pw=""+name.charAt(0)+(ph%10)+sum+sym[c]+name.charAt(len-1);

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

}

}

**OR**

import java.util.\*;

import java.io\*;

class p17

{

Public static void main(String() args[])

{

String roomno=”121”;

String phoneno=”9441331109”;

String username=”Millie”;

String password;

String a=” “,b=” “;

Int l=username.length();

A=phoneno.substring(0,1);

B=username.substring(l-2,l);

Password=roomno+a+b;

System.out.println(“password: “ +password);

}

}

OR

import java.util.\*;

import java.io.\*;

class prg2{

private static char[] GeneratingPassword(int length) {

String name,roomNo,phoneNo;

Scanner s=new Scanner(System.in);

System.out.println("enter the name:");

name=s.nextLine();

System.out.println("enter the room\_no:");

roomNo=s.nextLine();

System.out.println("enter the phone\_number:");

phoneNo=s.nextLine();

String specialChar="%$#@&";

Random random=new Random();

String combination=name+roomNo+phoneNo+specialChar;

char[] password=new char[length];

password[0]=name.charAt(random.nextInt(name.length()));

password[1]=roomNo.charAt(random.nextInt(roomNo.length()));

password[2]=phoneNo.charAt(random.nextInt(phoneNo.length()));

password[3]=specialChar.charAt(random.nextInt(specialChar.length()));

for(int i=4;i<length;i++) {

password[i]=combination.charAt(random.nextInt(combination.length()));

}

return password;

}

public static void main(String []args) {

System.out.println("program is to generate a random pasword for a given name,roomno,phone num:");

System.out.println("enter the length of password:");

Scanner s=new Scanner(System.in);

int length=s.nextInt();

System.out.println(GeneratingPassword(length));

}

}

**PART-B**

1. **Develop a small java application, which accepts employee id from the command prompt and displays the details using arrays**

**Emp No Emp Name Department Designation and Salary**

**Your may assume that the array is initialized with the following details**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Emp No.** | **Emp Name** | **Join Date** | **Desig Code** | **Dept** | **Basic** | **HRA** | **IT** |
| 1001 | Abc | 01/04/2009 | E | R&D | 20000 | 8000 | 3000 |
| 1002 | Opqr | 23/08/2012 | C | PM | 30000 | 12000 | 9000 |
| 1003 | Ghi | 12/11/2008 | K | Acct | 10000 | 8000 | 1000 |
| 1004 | Wxyz | 29/01/2013 | R | Front Desk | 12000 | 6000 | 2000 |
| 1005 | Jklmn | 16/07/2005 | M | Engg | 50000 | 20000 | 20000 |
| 1006 | Stuv | 01/01/2000 | E | Manufacturing | 23000 | 9000 | 4400 |
| 1007 | Def | 12/06/2006 | C | PM | 29000 | 12000 | 10000 |

Salary is calculated as Basic+HRA+DA-IT. (DA details are given in the Designation table)

Designation details:

|  |  |  |
| --- | --- | --- |
| **Designation Code** | **Designation** | **DA** |
| e | Engineer | 20000 |
| c | Consultant | 32000 |
| k | Clerk | 12000 |
| r | Receptionist | 15000 |
| m | Manager | 40000 |

Use Switch-Case to print Designation in the output and to find the value of DA for a particular employee

Employee.java

import java.util.Date;

import java.util.Stack;

public class Employee

{

public static void main(String[] args)

{

System.out.println("Enter Valid Employee ID : \n");

int[] EmpId={1001,1002,1003,1004,1005,1006,1007};

String[] EmpName={"Abc","Opqr","Ghi","Wxyz","Jklmn","Stuv","Def"};

String[] JoinDate={"01/04/2009","23/08/2012","12/11/2008","29/01/2013","16/07/2005","01/01/2000","12/06/2006"};

char[] DesigCode={'e','c','k','r','m','e','c'};

String[] Department={"R&D","PM","Acct","Front Desk","Engg","Manufacturing","PM"};

double[] Basic={20000,30000,10000,12000,50000,23000,29000};

double[] HRA={8000,12000,8000,6000,20000,9000,12000};

double[] IT={3000,9000,1000,2000,20000,4400,10000};

char[] DesignationCode={'e','c','k','r','m'};

String[] Designation={"Engineer","Consultant","Clerk","Receptionist","Manager"};

double[] DA={20000,32000,12000,15000,40000};

int flag=0;

int id=Integer.parseInt(args[0]);

for(int i=0;i<EmpId.length;i++)

{

if(EmpId[i]==id)

{

flag=1;

System.out.println("Emp Id. Emp Name Department Designation DA");

System.out.print(EmpId[i]+" "+EmpName[i]+" "+Department[i]);

for(int j=0;j<DesignationCode.length;j++)

{

if(DesigCode[i]==DesignationCode[j])

{ System.out.print(" "+Designation[j]+" ");

double sum=Basic[i]+HRA[i]+DA[j]-IT[i];

System.out.print(sum);

}

}

}

}

if(flag==0)

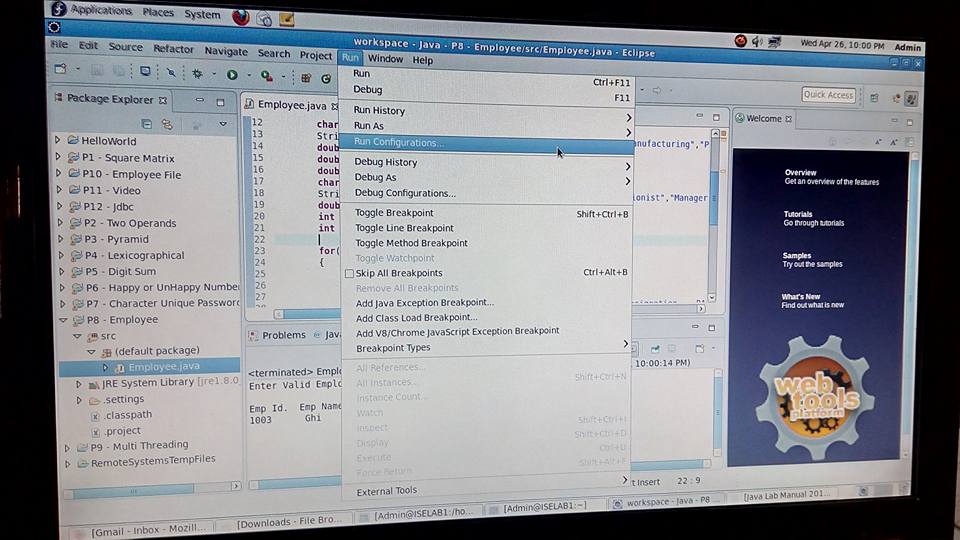
System.out.println("There is no employee with EmpId : " +id);

}

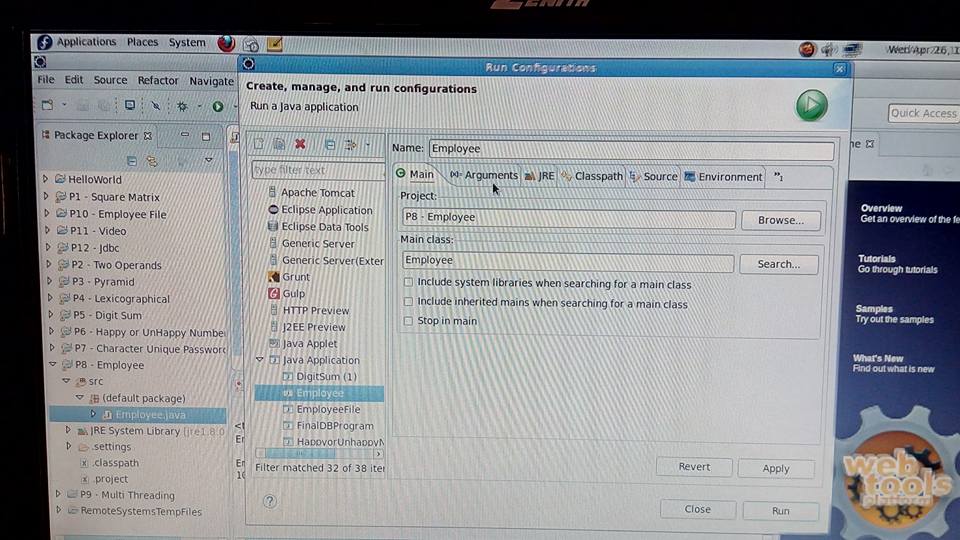
}

Output

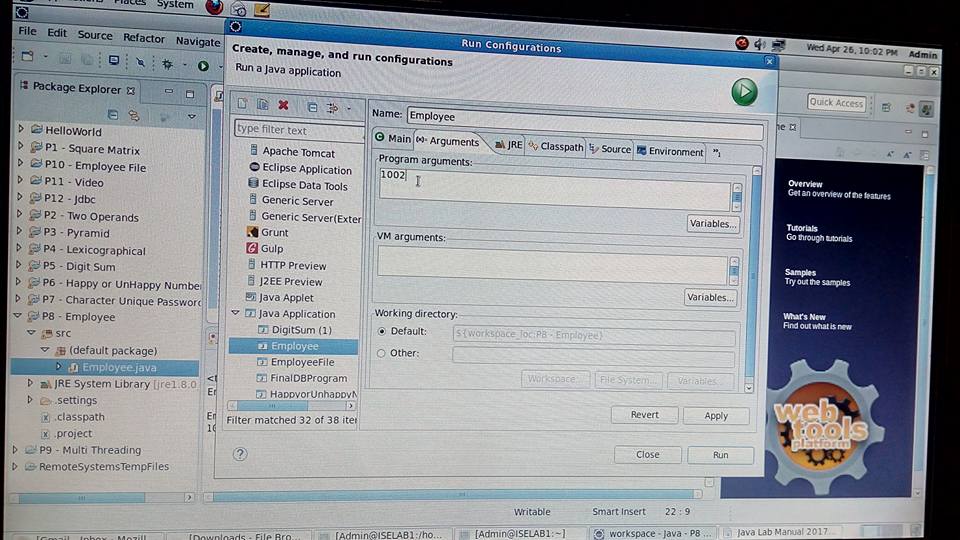
1. Run -> Run Configurations



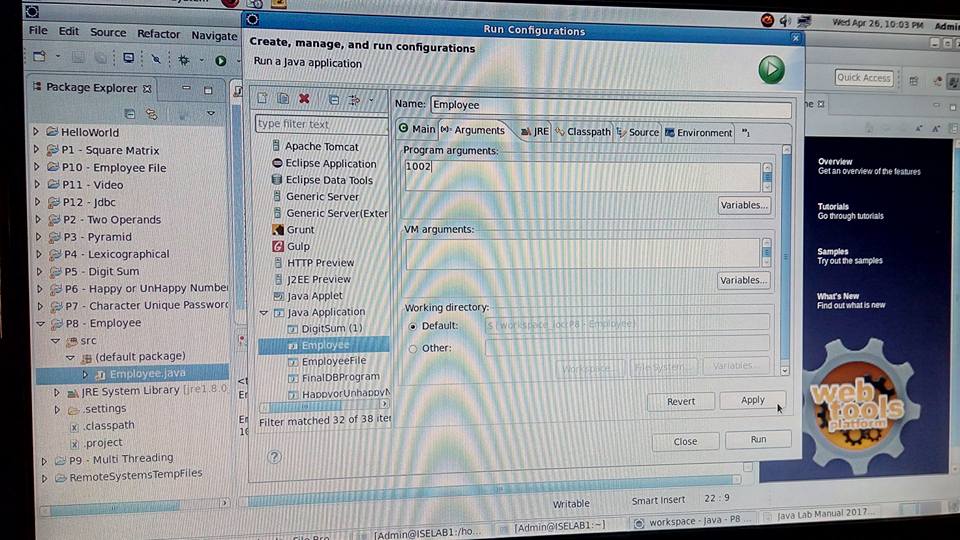
2. Select Arguments



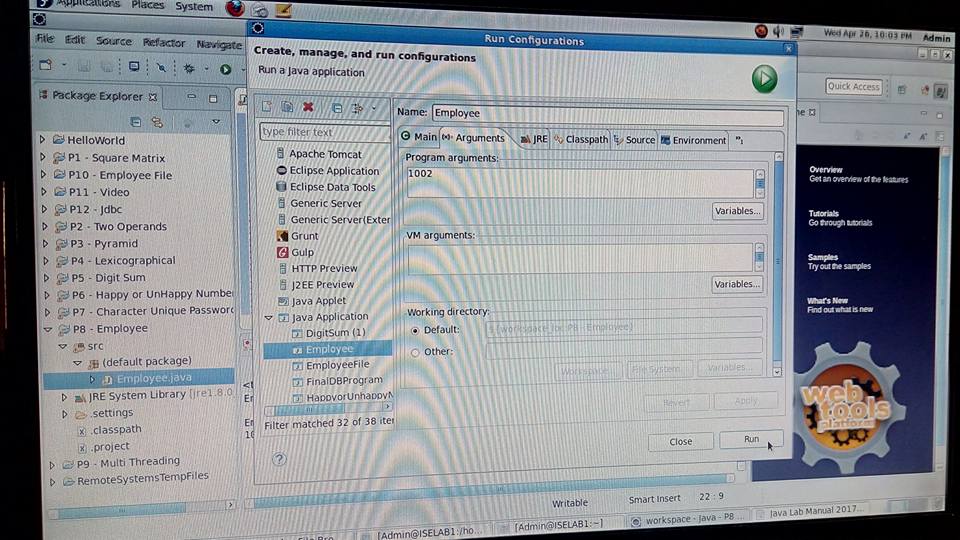
3. Enter Valid Employee ID :

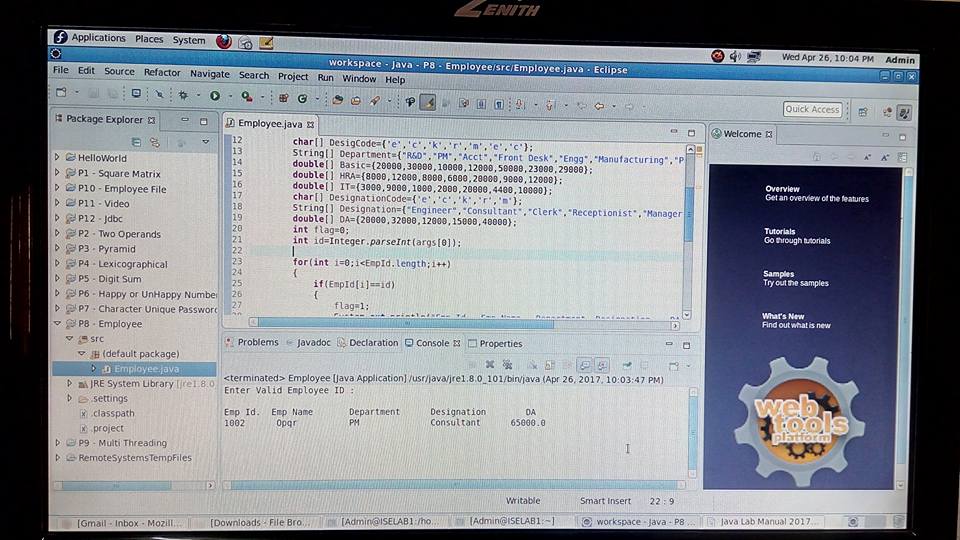


4. Apply



5. Run





1. **Develop a producer-consumer problem using the concept of multithreading**

**import** java.lang.Thread;

**class** Producer **extends** Thread {

StringBuffer buffer;

**boolean** dp = **false**;

Producer()

{

buffer = **new** StringBuffer(4);

}

**public** **void** run()

{

**synchronized** (buffer)

{

**for** (**int** i = 0; i < 4; i++) {

**try** {

buffer.append(i);

System.***out***.println("Produced " + i);

}

**catch** (Exception e) {

e.printStackTrace();

}

}

System.***out***.println("Buffer is FUll");

buffer.notify();

}

}

}

**class** Consumer **extends** Thread {

Producer p;

Consumer(Producer temp)

{

p = temp;

}

**public** **void** run()

{

**synchronized** (p.buffer)

{

**try** {

p.buffer.wait();

}

**catch** (Exception e) {

e.printStackTrace();

}

**for** (**int** i = 0; i < 4; i++) {

System.***out***.print(p.buffer.charAt(i) + " ");

}

System.***out***.println("\nBuffer is Empty");

}

}

}

**class** Main{

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

{

Producer p = **new** Producer();

Consumer c = **new** Consumer(p);

Thread t1 = **new** Thread(p);

Thread t2 = **new** Thread(c);

t2.start();

t1.start();

}

}

OR

import java.util.\*;

import java.io.\*;

public class prg2 {

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

final PC pc=new PC();

Thread t1=new Thread(new Runnable(){

//@override

public void run() {

try {

pc.produce();

}

catch(InterruptedException e) {

e.printStackTrace();

}

}

});

Thread t2=new Thread(new Runnable(){

//@override

public void run() {

try {

pc.consume();

}

catch(InterruptedException e) {

e.printStackTrace();

}

}

});

t1.start();

t2.start();

t1.join();

t2.join();

}

public static class PC{

LinkedList<Integer> list=new LinkedList<>();

int capacity=2;

public void produce() throws InterruptedException{

int value=0;

while(true) {

synchronized(this) {

while(list.size()==capacity)

wait();

System.out.println("Producer Produced-"+value);

list.add(value++);

notify();

Thread.sleep(1000);

}

}

}

public void consume() throws InterruptedException{

int value=0;

while(true) {

synchronized(this) {

while(list.size()==0)

wait();

int val=list.removeFirst();

System.out.println("Consumer consumed-"+val);

notify();

Thread.sleep(1000);

}

}

}

}

}

1. **Design and Implement GUI for managing Employee Details using concepts of Files**

EmployeeFile.java

import java.awt.Color;

import java.awt.Dimension;

import java.awt.GridLayout;

import java.awt.event.ActionEvent;

import java.awt.event.ActionListener;

import java.io.File;

import java.io.FileWriter;

import java.io.IOException;

import javax.swing.\*;

public class EmployeeFile

{

private static Color black;

public static void main(String[] args)

{

JFrame frameobj = new JFrame(); //creating frame

frameobj.setSize(500, 500); //declaring frame size

GridLayout g1=new GridLayout(5,2); // layout of the frame

frameobj.setLayout(g1); //layout is set to the frame created

JPanel p1=new JPanel(); //creating panels

JPanel p2=new JPanel();

JPanel p3=new JPanel();

JPanel p4=new JPanel();

JPanel p5=new JPanel();

JPanel p6=new JPanel();

JPanel p7=new JPanel();

JPanel p8=new JPanel();

JPanel p9=new JPanel();

JPanel p10=new JPanel();

JLabel l1=new JLabel("NAME"); //creating labels

JLabel l2=new JLabel("ID");

JLabel l3=new JLabel("DOJ");

JLabel l4=new JLabel("DOB");

JTextField f1=new JTextField(); //create obj for txtfield

JTextField f2=new JTextField();

JTextField f3=new JTextField();

JTextField f4=new JTextField();

f1.setPreferredSize(new Dimension(200,30)); //size of txtfield

f2.setPreferredSize(new Dimension(200,30));

f3.setPreferredSize(new Dimension(200,30));

f4.setPreferredSize(new Dimension(200,30));

JButton b1=new JButton("SUBMIT");

JButton b2=new JButton("RESET");

b1.addActionListener(new ActionListener()

{

@Override

public void actionPerformed(ActionEvent e)

{

File fileobj=new File("/home/Admin/workspace/P10 - Employee File/File.txt");

try

{

FileWriter fw=new FileWriter(fileobj.getAbsoluteFile(),true);

System.out.println("\n NAME : " +f1.getText() +"\n" +"ID : " +f2.getText() +"\n" +"DOJ : " +f3.getText() +"\n" +"DOB : "+f4.getText() +"\n");

fw.write("\n NAME : " +f1.getText() +"\n" +"ID : " +f2.getText() +"\n" +"DOJ : " +f3.getText() +"\n" +"DOB : "+f4.getText() +"\n");

fw.close();

}

catch (IOException e1)

{

e1.printStackTrace();

}

}

});

b2.addActionListener(new ActionListener()

{

@Override

public void actionPerformed(ActionEvent e)

{

f1.setText(" ");

f2.setText(null);

f3.setText(null);

f4.setText(null);

}

});

p1.add(l1); //add labels to panels where labels=name,id,doj,dob

p3.add(l2);

p5.add(l3);

p7.add(l4);

p2.add(f1); //add textfield to panels where txtfield is user defined

p4.add(f2);

p6.add(f3);

p8.add(f4);

p9.add(b1); //add buttons to panel

p10.add(b2);

//l1.setBorder(BorderFactory.createLineBorder(Color.black));

l1.setBorder(BorderFactory.createLineBorder(black,10));

l2.setBorder(BorderFactory.createLineBorder(black,10));

l3.setBorder(BorderFactory.createLineBorder(black,10));

l4.setBorder(BorderFactory.createLineBorder(black,10));

frameobj.add(p1); //add panels to frames

frameobj.add(p2);

frameobj.add(p3);

frameobj.add(p4);

frameobj.add(p5);

frameobj.add(p6);

frameobj.add(p7);

frameobj.add(p8);

frameobj.add(p9);

frameobj.add(p10);

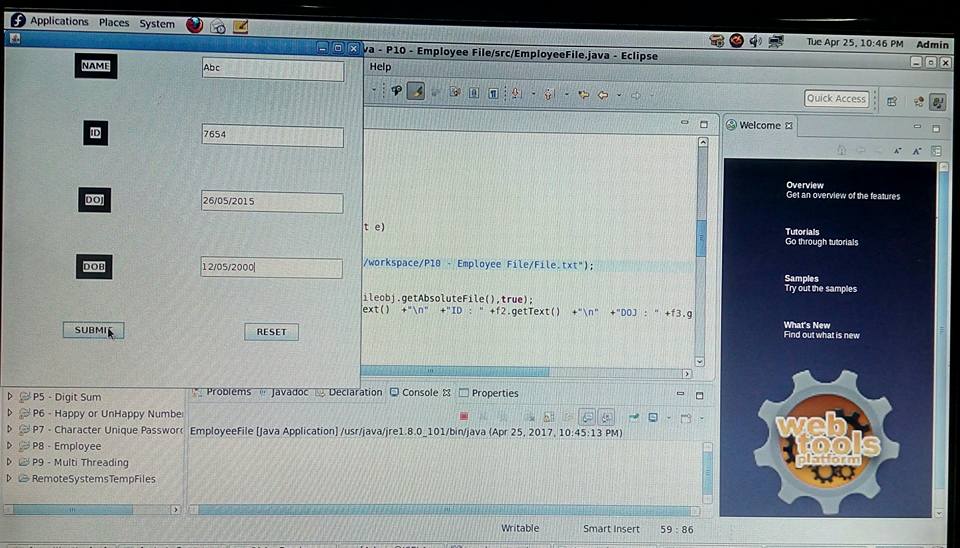
frameobj.setVisible(true);

}

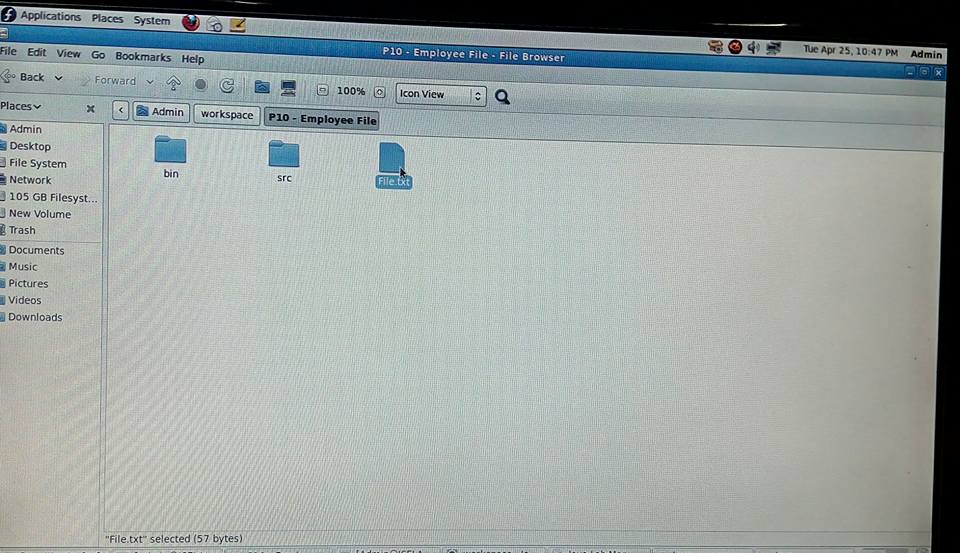
}

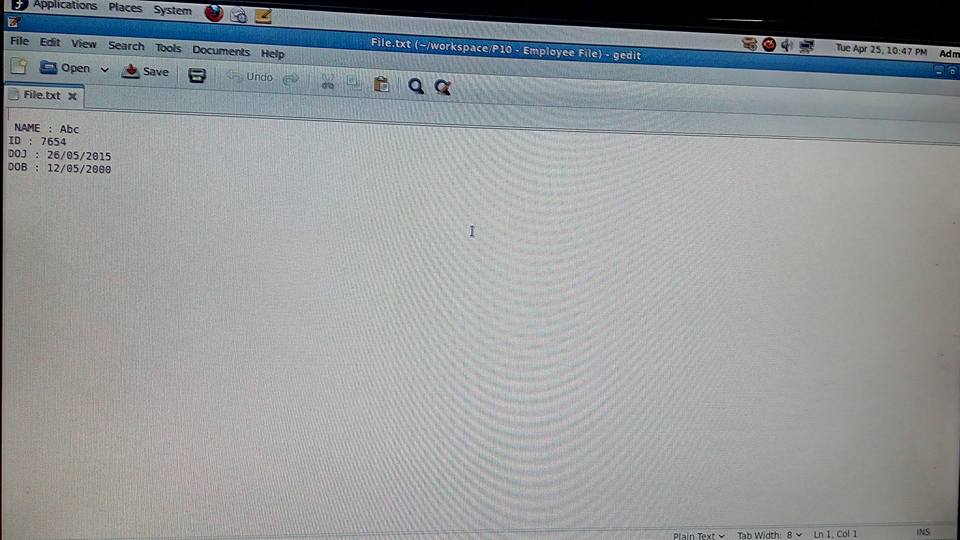
Output

1. Run



2. Go to folder & open file





Console Output

NAME : Abc

ID : 7654

DOJ : 26/05/2015

DOB : 12/05/2000

Note : Same output should be written into the file

1. **Write a program to implement different exception handling methods in java.**

**import** java.util.\*;

**public** **class** multexception {

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

**try** {

**int** arr[]=**new** **int**[7];

arr[5]=30/0;

}

**catch**(ArithmeticException e) {

System.***out***.println("Cannot divide a number by zero");

}

**catch**(ArrayIndexOutOfBoundsException e) {

System.***out***.println("Out of bound");

}

**catch**(Exception e) {

System.***out***.println("Other exception");

}

System.***out***.println("Out of try-catch block");

}

}

1. **Design a Tic-Tac toe game in swing**

**package** lab;

**import** javax.swing.\*;

**import** java.awt.\*;

**import** java.awt.event.\*;

**import** javax.swing.JOptionPane;

**public** **class** Tictactoe **extends** JFrame **implements** ActionListener{

**boolean** turn;

**static** **int** *i*,*j*;

JFrame f=**new** JFrame("TIC TAC TOE");

**private** JButton[][] b;

// String [][]matrix=new String[3][3];

JDialog d=**new** JDialog(f,"dialog box");

JLabel l=**new** JLabel();

JPanel p=**new** JPanel(**new** BorderLayout());

Tictactoe(){

p.setLayout(**new** GridLayout(3,3));

b =**new** JButton[3][3];

{

**for** (**int** row = 0; row < b.length; row++) {

**for** (**int** col = 0; col < b[0].length; col++) {

JButton cell = **new** JButton();

b[row][col] = cell;

cell.setPreferredSize(**new** Dimension(50,50));

p.add(cell);

//cell.setBounds((100+50\*col),(100+50\*row),50,50);

cell.addActionListener(**this**);

}

}

}

//JButton cell=new JButton();

//b[2][2]=cell;

//f.add(cell);

//cell.setBounds(200,200,50,50);

//cell.addActionListener(this);

f.add(p);

f.pack();

f.setVisible(**true**);

f.setSize(400,400);

f.setLayout(**null**);

//for(;i<3&&j<3;) {

// b[i][j].addActionListener(this);

//}

}

@Override

**public** **void** actionPerformed(ActionEvent e) {

**int** x,y;

JButton now=(JButton) e.getSource();

**if**(turn) {

**if**(!now.getText().equals("x")) {

now.setText("o");

turn=**false**;

}

}

**else** {

**if**(!now.getText().equals("o")) {

now.setText("x");

turn=**true**;

}

}

**for**(x=0,y=0;x<3;x++) {

**if**(((b[x][y].getText().equals("x"))&&(b[x][y+1].getText().equals("x"))&&(b[x][y+2].getText().equals("x")))){

l.setText("x won the game");

JOptionPane.*showMessageDialog*(f, "x won the game");

f.add(d);

d.add(l);

d.setSize(300,300);

d.setVisible(**true**);

}

}

**for**(y=0,x=0;y<3;y++) {

**if**(((b[x][y].getText().equals("x"))&&(b[x+1][y].getText().equals("x"))&&(b[x+2][y].getText().equals("x")))){

l.setText("x won the game");

JOptionPane.*showMessageDialog*(f, "x own the game");

f.add(d);

d.setSize(300,300);

d.setVisible(**true**);

}

}

y=0;

**if**(((b[x][y].getText().equals("x"))&&(b[x+1][y+1].getText().equals("x"))&&(b[x+2][y+2].getText().equals("x")))) {

l.setText("x won the game");

JOptionPane.*showMessageDialog*(f, "x won the game");

f.add(d);

d.setSize(300,300);

d.setVisible(**true**);

}

**if**(((b[0][2].getText().equals("x"))&&(b[1][1].getText().equals("x"))&&(b[2][0].getText().equals("x")))) {

l.setText("x won the game");

JOptionPane.*showMessageDialog*(f, "x won the game");

f.add(d);

d.setSize(300,300);

d.setVisible(**true**);

}

**for**(x=0,y=0;x<3;x++) {

**if**(((b[x][y].getText().equals("o"))&&(b[x][y+1].getText().equals("o"))&&(b[x][y+2].getText().equals("o")))){

l.setText("o won the game");

JOptionPane.*showMessageDialog*(f, "o won the game");

f.add(d);

d.setSize(300,300);

d.setVisible(**true**);

}

}

**for**(y=0,x=0;y<3;y++) {

**if**(((b[x][y].getText().equals("o"))&&(b[x+1][y].getText().equals("o"))&&(b[x+2][y].getText().equals("o")))){

l.setText("o won the game");

JOptionPane.*showMessageDialog*(f, "o won the game");

f.add(d);

d.setSize(300,300);

d.setVisible(**true**);

}

}

x=0;y=0;

**if**(((b[x][y].getText().equals("o"))&&(b[(x+1)][(y+1)].getText().equals("o"))&&(b[(x+2)][(y+2)].getText().equals("o")))) {

l.setText("o won the game");

JOptionPane.*showMessageDialog*(f,"o won the game");

f.add(d);

d.setSize(300,300);

d.setVisible(**true**);

}

**if**(((b[0][2].getText().equals("o"))&&(b[1][1].getText().equals("o"))&&(b[2][0].getText().equals("o")))) {

l.setText("o won the game");

JOptionPane.*showMessageDialog*(f,"o won the game");

f.add(d);

d.setSize(300,300);

d.setVisible(**true**);

}

}

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

**new** Tictactoe();

}

}

1. **Design and implement a simple inventory central system for a small video rental store using constructors and Object List**

Bean Class File : Video.java

**public** **class** Video

{

String mName;

**boolean** status;

**double** rating;

**public** Video(String mName, **boolean** status, **double** rating)

{

**super**();

**this**.mName = mName;

**this**.status = status;

**this**.rating = rating;

}

**public** String getmName()

{

**return** mName;

}

**public** **void** setmName(String mName)

{

**this**.mName = mName;

}

**public** **boolean** isStatus()

{

**return** status;

}

**public** **void** setStatus(**boolean** status)

{

**this**.status = status;

}

**public** **double** getRating()

{

**return** rating;

}

**public** **double** setRating(**double** rating)

{

**return** **this**.rating = rating;

}

}

Methods Class File : VideoMethods.java

**import** java.util.List;

**import** java.util.Scanner;

**import** java.util.ArrayList;

**public** **class** VideoMethods

{

List<Video> MovieList = **new** ArrayList<Video>();

**public** **void** AddMovies()

{

Scanner in=**new** Scanner(System.***in***);

System.***out***.print("Enter the name of the movie:");

String mName=in.nextLine();

System.***out***.print("Enter the status of the movie(True/False):");

**boolean** status=in.nextBoolean();

System.***out***.print("Enter the ratings for the movie(0-5):");

**double** rating=in.nextDouble();

Video v=**new** Video(mName, status, rating);

MovieList.add(v);

System.***out***.println("Library Initialized");

}

**public** **void** DisplayAll()

{

**if**(MovieList.isEmpty())

{

System.***out***.println("No movies in the library");

}

**for**(Video m : MovieList)

{

System.***out***.println("Movie : " +m.getmName()+" "+"Status : "+m.isStatus()+" "+"Rating "+m.getRating());

}

}

**boolean** RentOut(String name)

{

**for**(Video m :MovieList)

{

**if**(m.getmName().equalsIgnoreCase(name))

{

**if**(m.isStatus())

{

m.setStatus(**false**);

**return** **true**;

}

}

**return** **false**;

}

**return** **false**;

}

**public** **void** CollectIn(String name,**double** rat)

{

**boolean** flag=**false**;

**for**(Video m :MovieList)

{

**if**(m.getmName().equalsIgnoreCase(name))

{

m.setStatus(**true**);

flag=**true**;

Math.*round*(m.setRating((m.getRating() + rat)/2));

}

}

**if**(!flag)

{

System.***out***.println("Requested Movie not rented out");

}

}

}

Main Class File: VideoMain.java

**import** java.util.Scanner;

**public** **class** VideoMain

{

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

{

VideoMethods mm = **new** VideoMethods();

**while**(**true**)

{

System.***out***.println("%%%%%%%% VIDEO LIBRARY CENTER %%%%%%%%");

**int** n;

Scanner in = **new** Scanner(System.***in***);

System.***out***.println("1.ADD MOVIES");

System.***out***.println("2.DISPLAY MOVIES");

System.***out***.println("3.RENT OUT");

System.***out***.println("4.COLLECT BACK ");

System.***out***.println("PLEASE ENTER YOUR OPTION");

n = in.nextInt();

**switch**(n)

{

**case** 1:mm.AddMovies();

**break**;

**case** 2:mm.DisplayAll();

**break**;

**case** 3:System.***out***.print("Enter the movie you want to rent.");

in.nextLine();

**if**(mm.RentOut(in.nextLine()))

{

System.***out***.println("Rent out successfull");

}

**else**

{

System.***out***.println("Sorry!! Not Available");

}

**break**;

**case** 4:System.***out***.println("Enter the name and the ratings of the movie");

in.nextLine();

mm.CollectIn(in.nextLine(),in.nextDouble());

**break**;

}

}

}

}

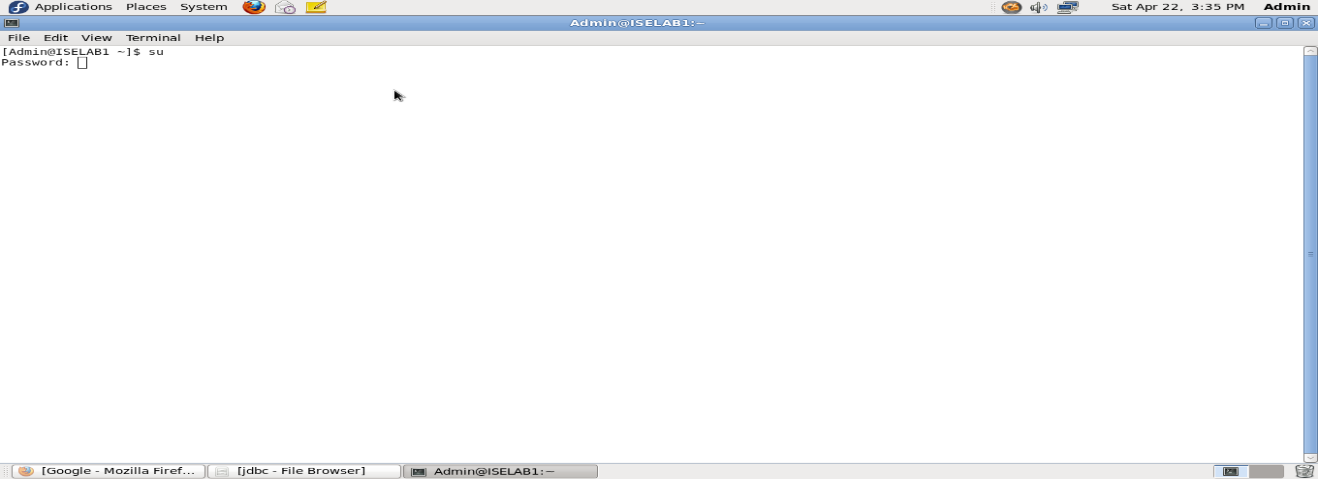
1. **Given the information about employees of an organization, develop a small java application, using JDBC.**

**Ex:**

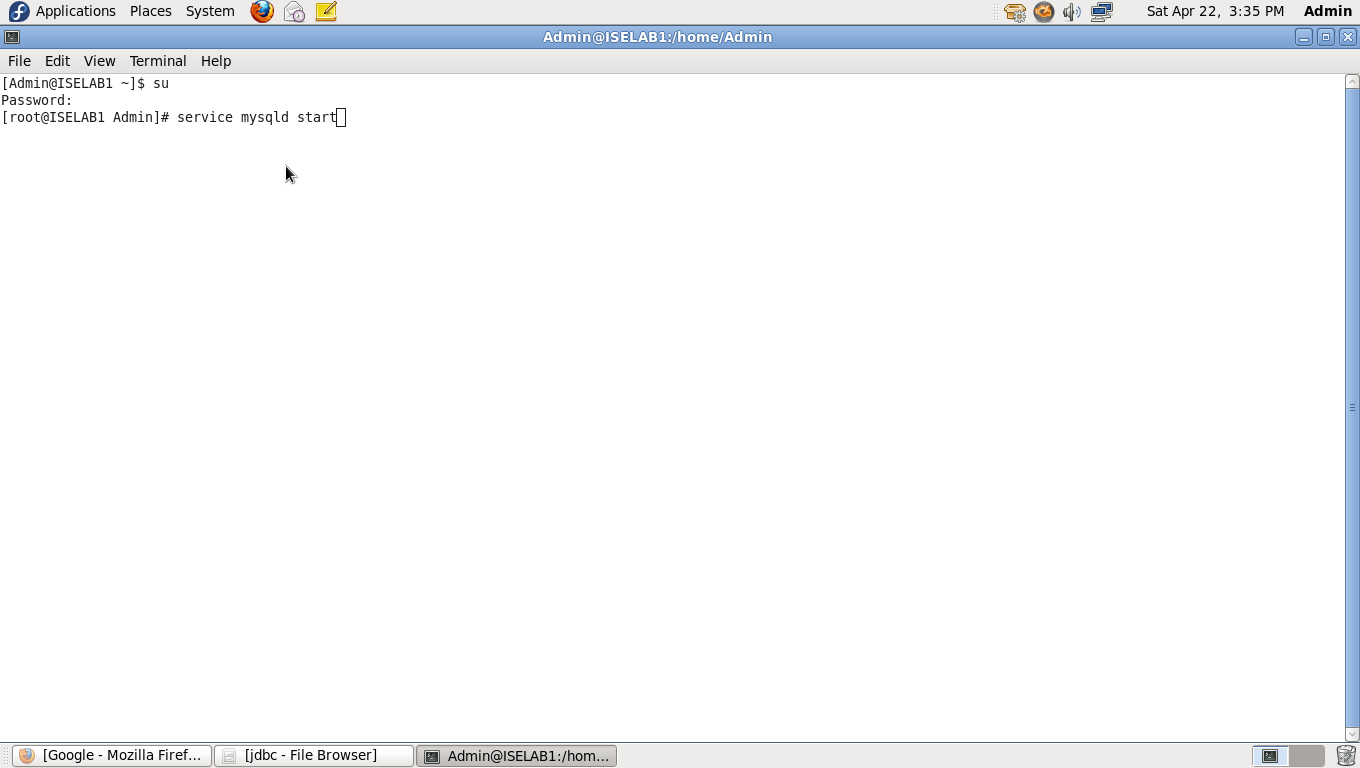
|  |  |  |  |
| --- | --- | --- | --- |
| **EmpName** | **Age** | **Dept** | **Salary** |
| DEF | 25 | ISE | 50000 |
| ABC | 30 | ISE | 55000 |

**How Create Mysql Database & Table**

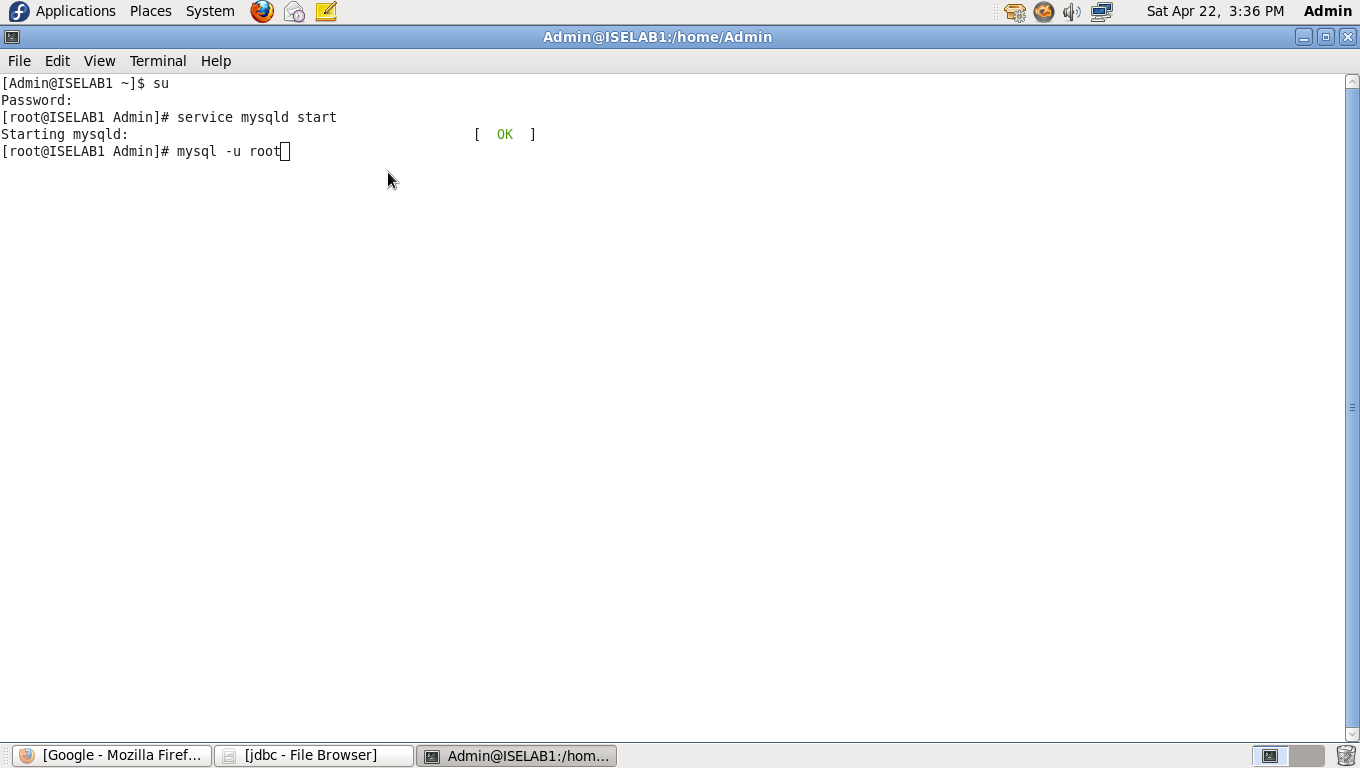
1. Open Terminal -> su (type root password)

****

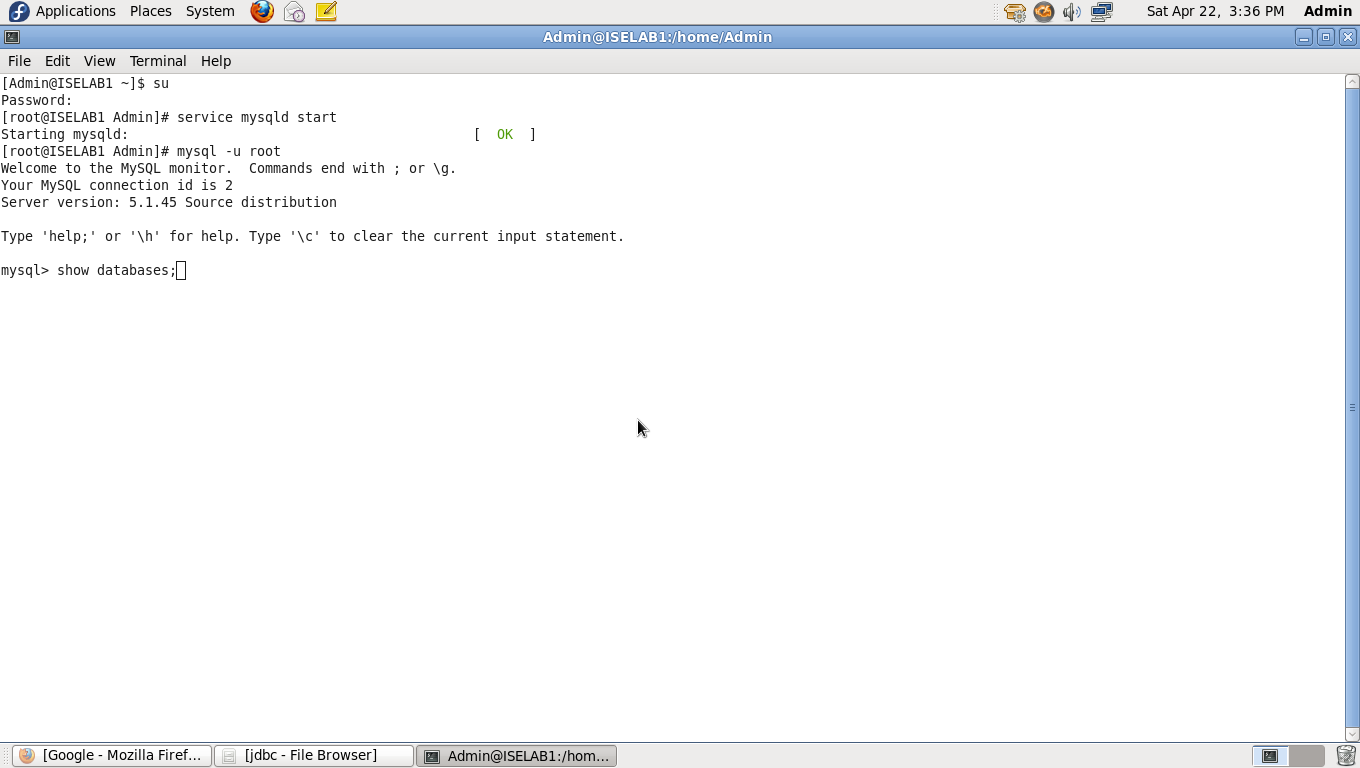
2. service mysqld start

****

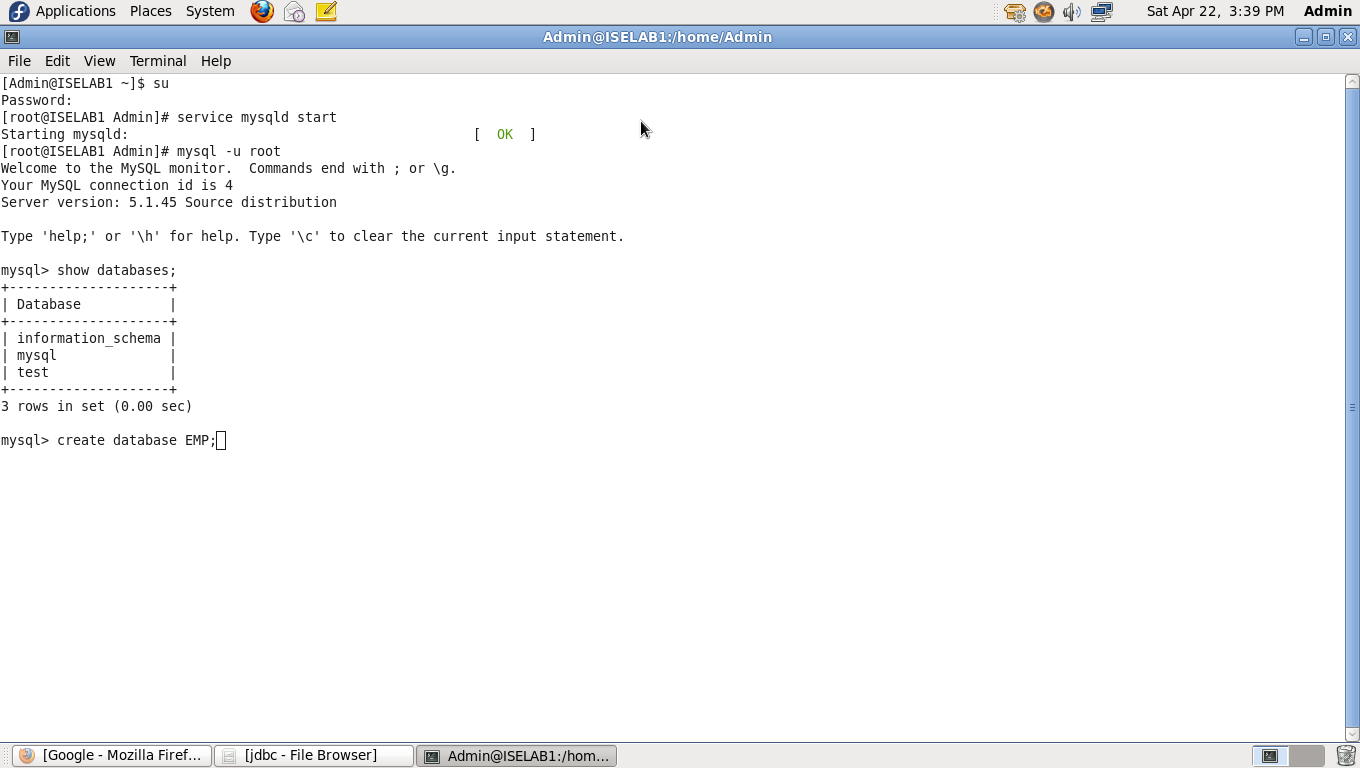
3. mysql –u root

****

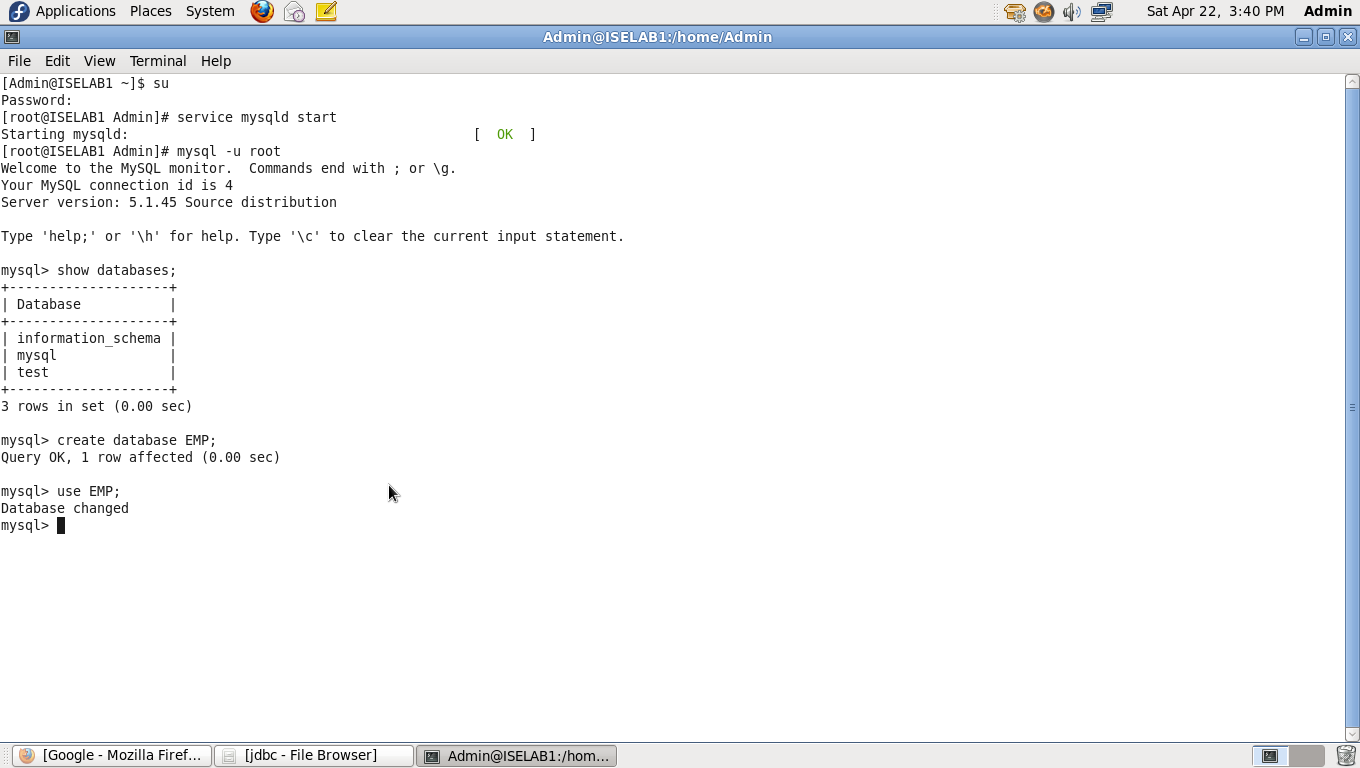
4. mysql> show databases;



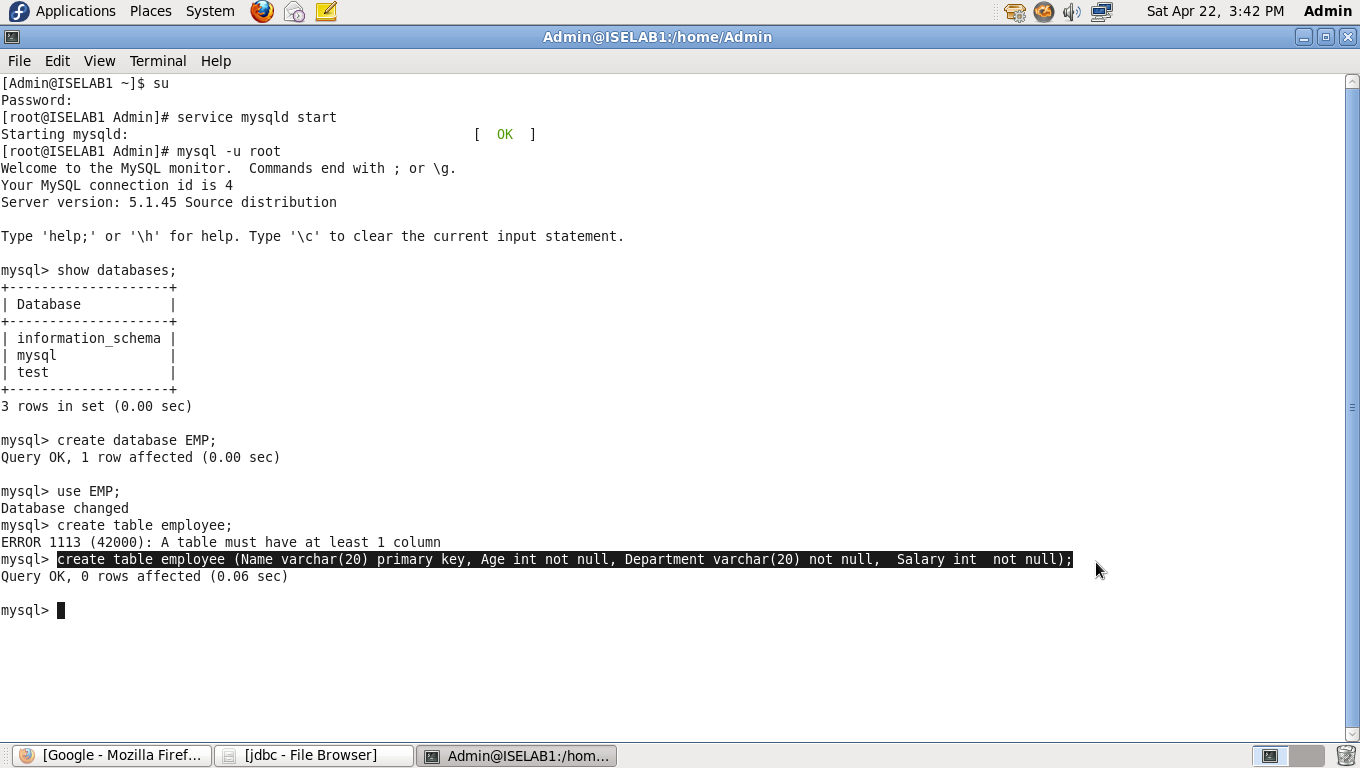
5. mysql> create database databasename;



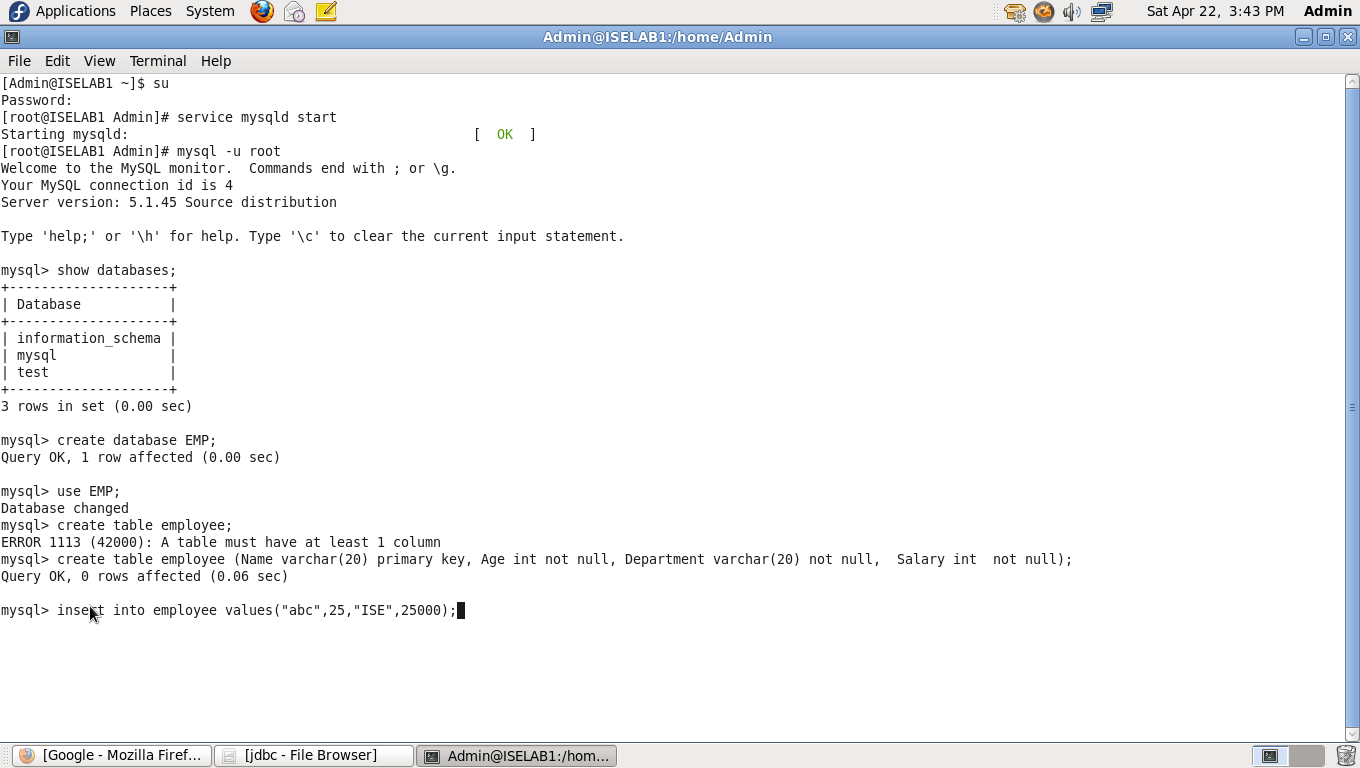
6. mysql> use databasename;



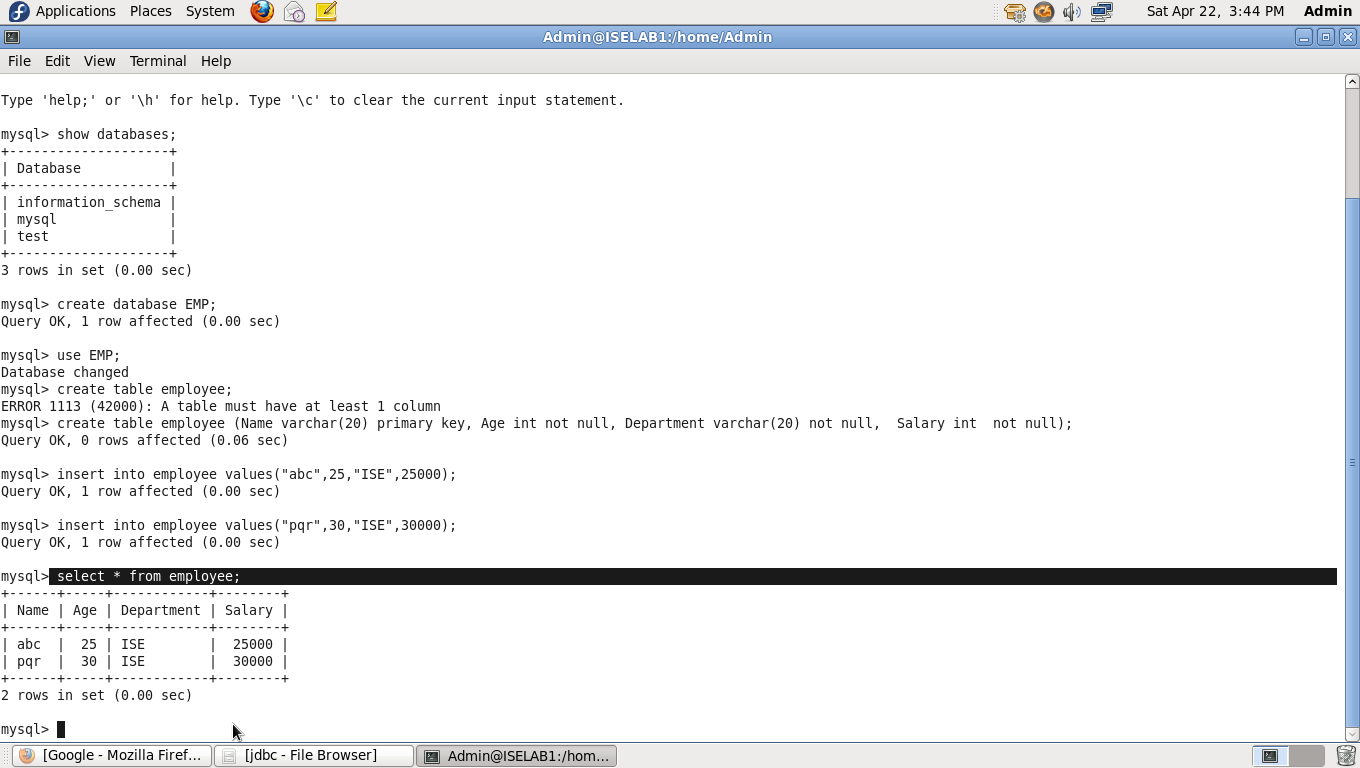
7. mysql> create a table and attributes



8. mysql> insert table values



9. Select Table



After creating database, minimize the mysql terminal and open eclipse.

Create new java project with any suitable name.

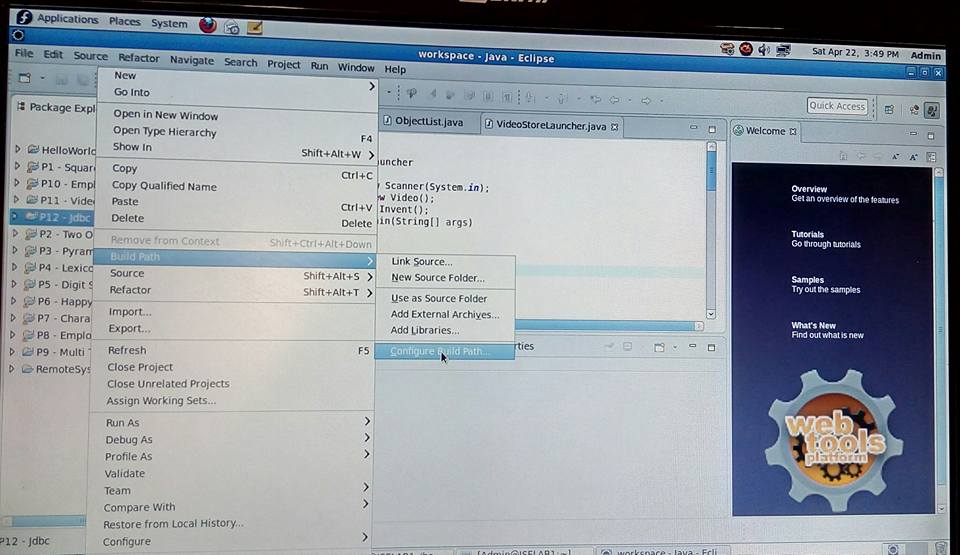
Right click on the project name from the project explorer and navigate to “build path”,

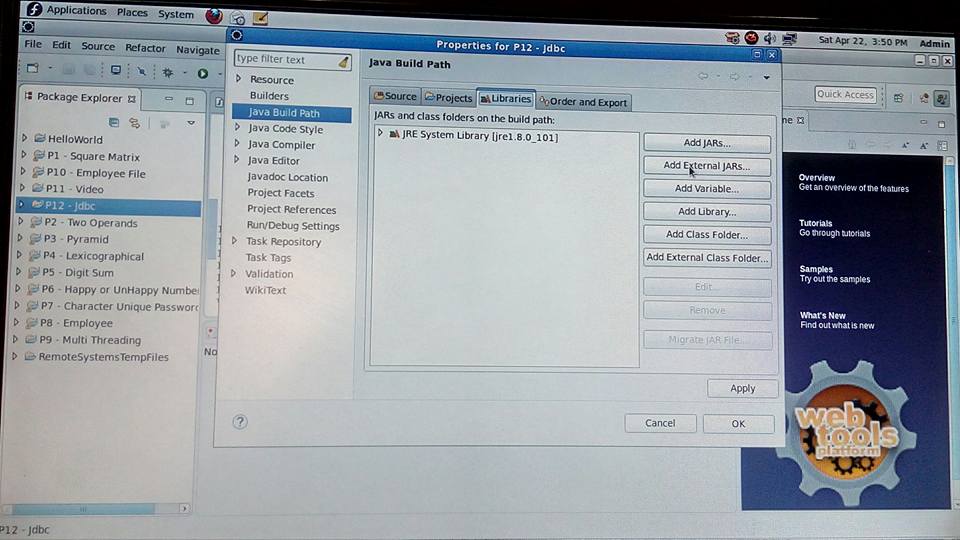
Inside build path sub tab, navigate to “Configure build path” and left click on it.

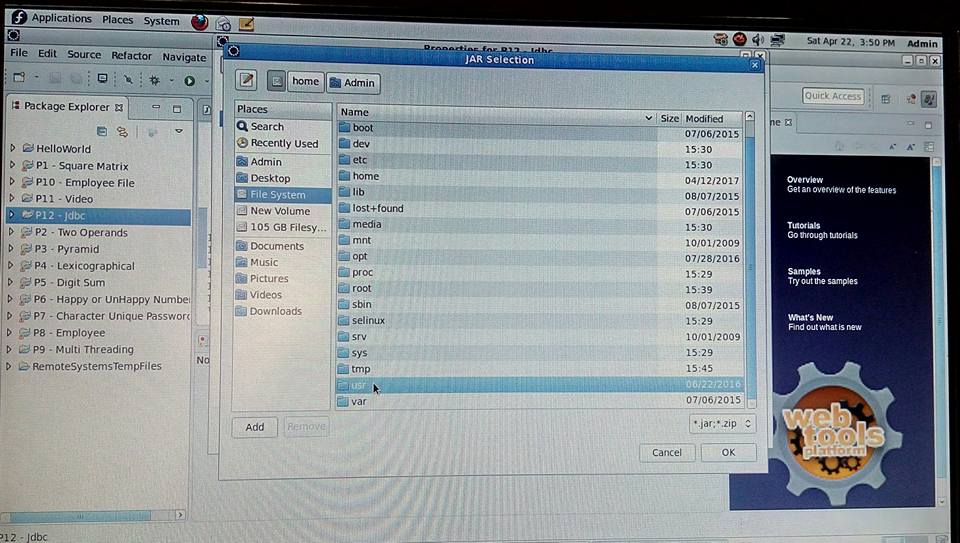
Select Libraries and click on Add external jars,

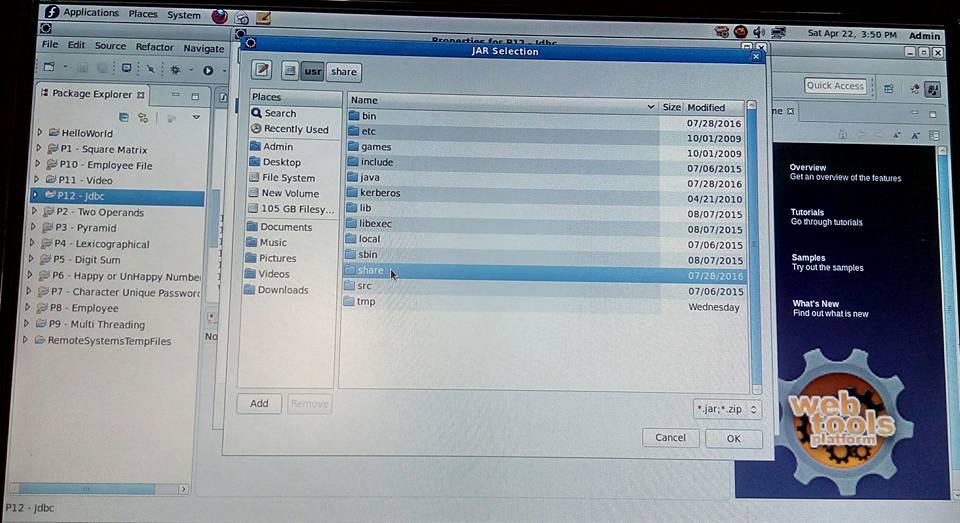
Navigate through your file system to usr/share/java/mysql-connector-java-5.1.12.jar. After successful selection, click on Apply followed by click on Ok buttons.

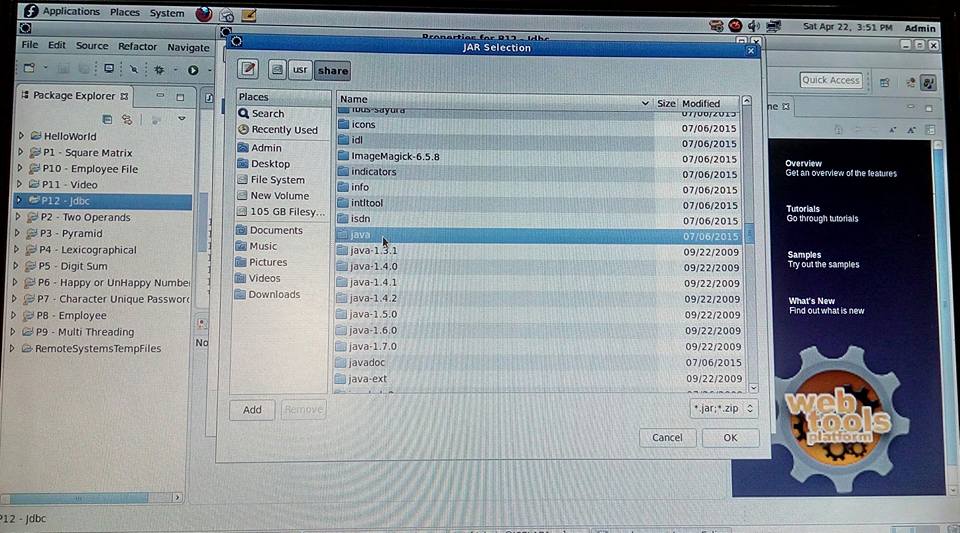
(Note : This step is important, otherwise program will not be able to find the Mysql Driver)

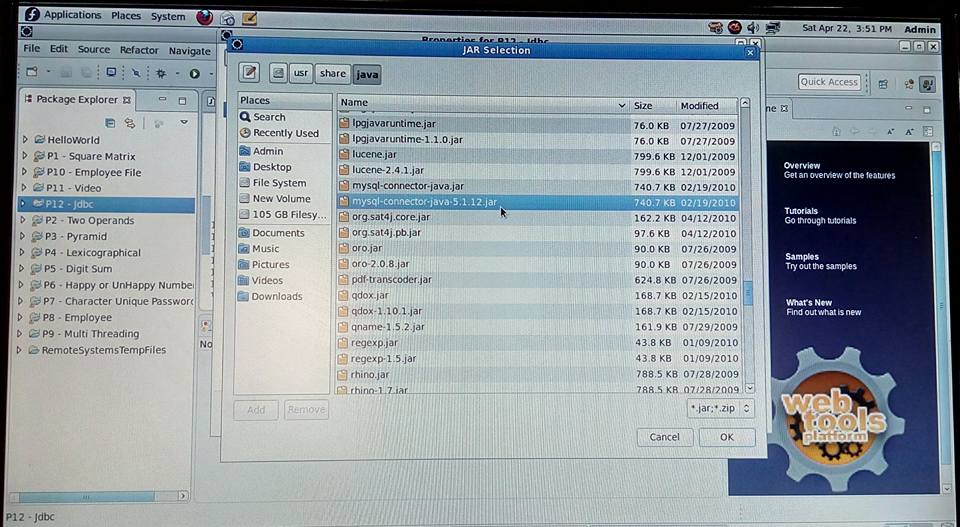


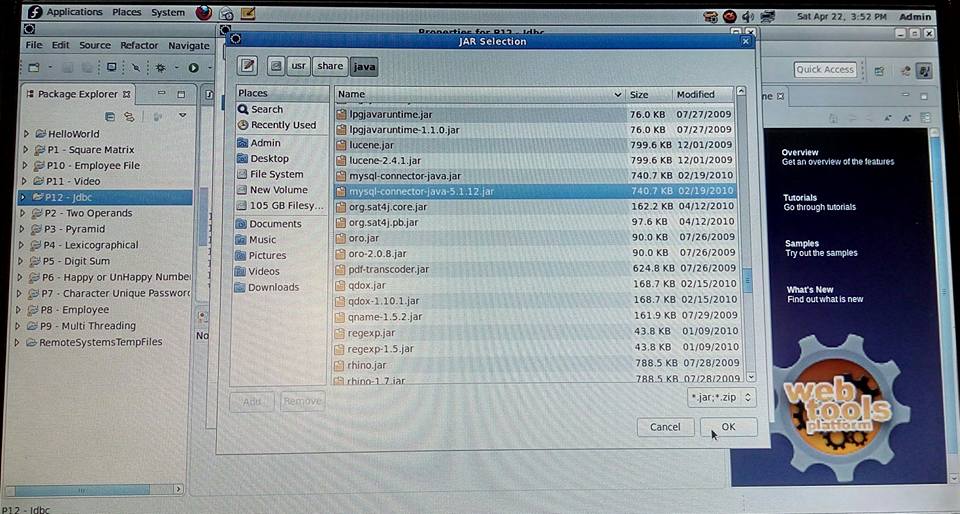


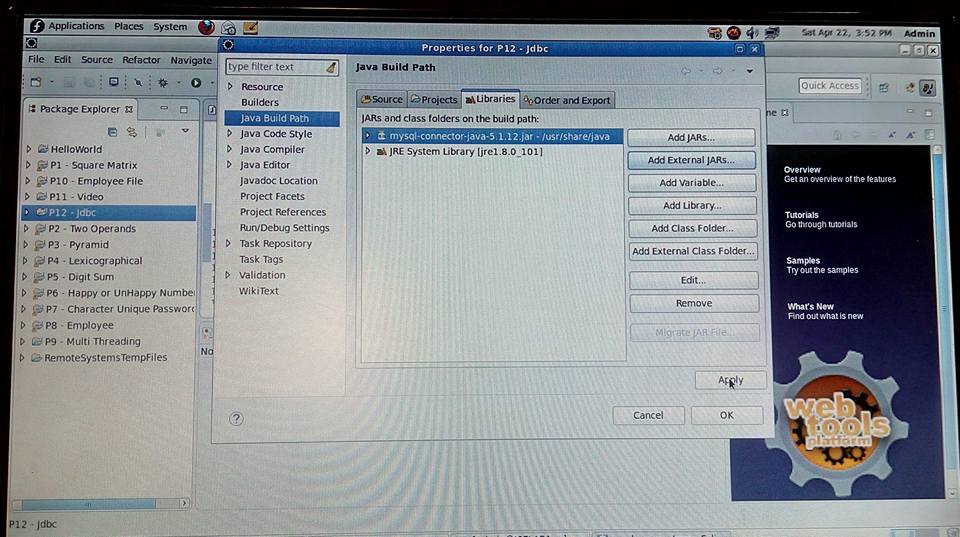


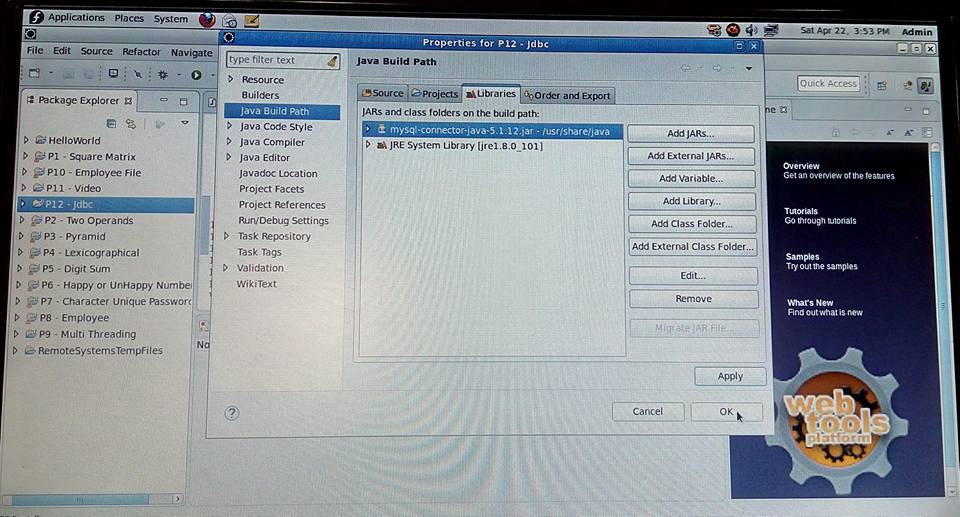












DBConnection.java

import java.sql.Connection;

import java.sql.DriverManager;

import java.sql.SQLException;

public class DBConnection

{

public Connection getDBconnection()

{

Connection conn=null;

try

{

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

System.out.println("Registered successfully");

}

catch (ClassNotFoundException e)

{

e.printStackTrace();

}

try

{

conn=DriverManager.getConnection("jdbc:mysql://localhost:3306/EMP","root","");

System.out.println("Connection successfull\n");

}

catch (SQLException e)

{

e.printStackTrace();

}

return conn;

}

}

Employee.java

**public** **class** Employee

{

String name;

**int** age;

String dept;

**double** sal;

**public** String getName()

{

**return** name;

}

**public** **void** setName(String name)

{

**this**.name = name;

}

**public** **int** getAge()

{

**return** age;

}

**public** **void** setAge(**int** age)

{

**this**.age = age;

}

**public** String getDept()

{

**return** dept;

}

**public** **void** setDept(String dept)

{

**this**.dept = dept;

}

**public** **double** getSal()

{

**return** sal;

}

**public** **void** setSal(**double** sal)

{

**this**.sal = sal;

}

}

DAO.java

import java.sql.Connection;

import java.sql.PreparedStatement;

import java.sql.ResultSet;

import java.sql.SQLException;

public class DAO

{

public int insertemp(Employee e)

{

int r=0;

Connection conn=new DBConnection().getDBconnection();

try

{

PreparedStatement pst=conn.prepareStatement("insert into employee values(?,?,?,?)");

pst.setString(1, e.getName());

pst.setInt(2, e.getAge());

pst.setString(3, e.getDept());

pst.setDouble(4, e.getSal());

r=pst.executeUpdate();

}

catch (SQLException e1)

{

e1.printStackTrace();

}

return r;

}

public int deleteemp(String emp)

{

int s=0;

Connection conn=new DBConnection().getDBconnection();

try

{

PreparedStatement pst=conn.prepareStatement("delete from employee where name=?");

pst.setString(1, emp);

s=pst.executeUpdate();

}

catch (SQLException e1)

{

e1.printStackTrace();

}

return s;

}

public void displayname(String ename)

{

Connection conn=new DBConnection().getDBconnection();

try

{

PreparedStatement pst = conn.prepareStatement("select \* from employee where name=?");

pst.setString(1, ename);

ResultSet rs=pst.executeQuery();

while(rs.next())

{

System.out.println("Name : " +rs.getString(1)+ "\t"+ "Age : " +rs.getInt(2)+ "\t"+ "Dept :" +rs.getString(3)+ "\t"+ "Salary :" +rs.getDouble(4));;

}

}

catch (SQLException e)

{

e.printStackTrace();

}

}

public void displayall()

{

Connection conn=new DBConnection().getDBconnection();

try

{

PreparedStatement pst=conn.prepareStatement("select \* from employee");

ResultSet rs=pst.executeQuery();

while(rs.next())

{

System.out.println("Name : " +rs.getString(1)+ "\t"+ "Age : " +rs.getInt(2)+"\t"+ "Dept :" +rs.getString(3)+"\t" + "Salary :" +rs.getDouble(4));;

}

}

catch (SQLException e1)

{

e1.printStackTrace();

}

}

}

FinalDBProgram.java

**import** java.util.Scanner;

**public** **class** FinalDBProgram

{

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

{

**for**( ; ; )

{

Scanner in=**new** Scanner(System.***in***);

System.***out***.println("\n 1. Insert Emp \n 2. Delete Emp \n 3. Display Acc to Name \n 4. Display All \n 5. Exit \n");

System.***out***.println("Enter your choice");

**int** n=in.nextInt();

Employee e=**new** Employee();

DAO d=**new** DAO();

**switch**(n)

{

**case** 1:

System.***out***.println("Enter the Employee Name : ");

e.setName(in.next());

System.***out***.println("Enter the Age : ");

e.setAge(in.nextInt());

System.***out***.println("Enter the Dept : ");

e.setDept(in.next());

System.***out***.println("Enter the Salary : ");

e.setSal(in.nextDouble());

d.insertemp(e);

System.***out***.println("Employee added successfully");

**break**;

**case** 2:

System.***out***.println("Enter the employee name : ");

String Newname=in.next();

d.deleteemp(Newname);

System.***out***.println("Employee deleeted successfully");

**break**;

**case** 3:System.***out***.println("Enter the Employee name to display its attributes : ");

String ename=in.next();

System.***out***.println("Employee details...");

d.displayname(ename);

**break**;

**case** 4:System.***out***.println("Employee deatils are as follows...");

d.displayall();

**break**;

**case** 5:System.*exit*(0);

**break**;

**default**:

System.***out***.println("Please Choose Valid option \n");

**break**;

}

}

}

}

## Program to find the maximum and minimum value node from a doubly linked list.

1. **public** **class** MinMax {
3. //Represent a node of the doubly linked list
5. **class** Node{
6. **int** data;
7. Node previous;
8. Node next;
10. **public** Node(**int** data) {
11. **this**.data = data;
12. }
13. }
15. //Represent the head and tail of the doubly linked list
16. Node head, tail = **null**;
18. //addNode() will add a node to the list
19. **public** **void** addNode(**int** data) {
20. //Create a new node
21. Node newNode = **new** Node(data);
23. //If list is empty
24. **if**(head == **null**) {
25. //Both head and tail will point to newNode
26. head = tail = newNode;
27. //head's previous will point to null
28. head.previous = **null**;
29. //tail's next will point to null, as it is the last node of the list
30. tail.next = **null**;
31. }
32. **else** {
33. //newNode will be added after tail such that tail's next will point to newNode
34. tail.next = newNode;
35. //newNode's previous will point to tail
36. newNode.previous = tail;
37. //newNode will become new tail
38. tail = newNode;
39. //As it is last node, tail's next will point to null
40. tail.next = **null**;
41. }
42. }
44. //MinimumNode() will find out minimum value node in the list
45. **public** **int** minimumNode() {
46. //Node current will point to head
47. Node current = head;
48. **int** min;
50. //Checks if list is empty
51. **if**(head == **null**) {
52. System.out.println("List is empty");
53. **return** 0;
54. }
55. **else** {
56. //Initially, min will store the value of head's data
57. min = head.data;
58. **while**(current != **null**) {
59. //If the value of min is greater than the current's data
61. //Then, replace the value of min with current node's data
63. **if**(min > current.data)
64. min = current.data;
65. current = current.next;
66. }
67. }
68. **return** min;
69. }
71. //MaximumNode() will find out maximum value node in the list
72. **public** **int** maximumNode() {
73. //Node current will point to head
74. Node current = head;
75. **int** max;
77. //Checks if list is empty
78. **if**(head == **null**) {
79. System.out.println("List is empty");
80. **return** 0;
81. }
82. **else** {
83. //Initially, max will store the value of head's data
84. max = head.data;
85. //If value of max is lesser than current's data
86. //Then, replace value of max with current node's data
87. **while**(current != **null**) {
88. **if**(current.data > max)
89. max = current.data;
90. current = current.next;
91. }
92. }
93. **return** max;
94. }
96. **public** **static** **void** main(String[] args) {
98. MinMax dList = **new** MinMax();
99. //Add nodes to the list
100. dList.addNode(5);
101. dList.addNode(7);
102. dList.addNode(9);
103. dList.addNode(1);
104. dList.addNode(2);
106. //Prints the minimum value node in the list
107. System.out.println("Minimum value node in the list: "+ dList.minimumNode());
108. //Prints the maximum value node in the list
109. System.out.println("Maximum value node in the list: "+ dList.maximumNode());
110. }
111. }