1.

abstract class bank{

int Bank\_id;

String bank\_name, IFSC\_code;

bank(int id, String name, String IFSC){

Bank\_id = id;

bank\_name = name;

IFSC\_code = IFSC;

}

abstract void getROI();

void display(){

System.out.println("Bank\_id= "+Bank\_id+", bank\_name= "+bank\_name+", IFSC\_code = " +IFSC\_code);

}

}

class SBI extends bank{

int fd;

SBI(){

super(12, "SBI", "sbi394");

}

int roi = 5;

void getROI(){

System.out.println("ROI for SBI is "+roi +"%");

}

double total(int fd){

double sum =0;

sum = (fd \*roi)/100;

return sum;

}

}

class HDFC extends bank{

HDFC(){

super(24, "HDFC", "hdfc435");

}

int roi = 8;

void getROI(){

System.out.println("ROI for HDFC is "+roi +"%");

}

double total(int fd){

double sum =0;

sum = (fd \*roi)/100;

return sum;

}

}

class PNB extends bank{

PNB(){

super(46, "PNB", "pnb75984");

}

int roi = 6;

void getROI(){

System.out.println("ROI for PNB is "+roi +"%");

}

double total(int fd){

double sum =0;

sum = (fd \*roi)/100;

return sum;

}

}

class main{

static int fd = 500000;

public static void main(String args[]){

double gt;

String g;

SBI first = new SBI();

first.display();

HDFC sec = new HDFC();

sec.display();

PNB trd = new PNB();

trd.display();

first.getROI();

sec.getROI();

trd.getROI();

if (first.total(fd)>sec.total(fd)){

gt =first.total(fd);

g = "SBI";

}

else{

gt = sec.total(fd);

g = "HDFC";

}

if (gt<trd.total(fd)){

System.out.println("PNB is better");

}

else{

System.out.println(" Better profit is with "+g);

}

}

}

OUTPUT:

Bank\_id= 12, bank\_name= SBI, IFSC\_code = sbi394

Bank\_id= 24, bank\_name= HDFC, IFSC\_code = hdfc435

Bank\_id= 46, bank\_name= PNB, IFSC\_code = pnb75984

ROI for SBI is 5%

ROI for HDFC is 8%

ROI for PNB is 6%

Better profit is with HDFC

2.

abstract class EMP{

int emp\_id;

String name, date;

void get\_info(int id, String name, String join\_date)

{

emp\_id = id;

this.name = name;

this.date = join\_date;

}

abstract double salary();

void display()

{

double s = this.salary();

System.out.println("Salary is Rs " + s);

System.out.println("Details: ");

System.out.println("ID: " + emp\_id);

System.out.println("Name: " + name);

System.out.println("Date of joining: " + date);

}

}

class Manager extends EMP{

double salary()

{

return 65000 \* 1.75;

}

}

class Sales extends EMP{

double salary()

{

return 30000 \* 1.99;

}

}

public class p2 {

public static void main(String[] args) {

Manager mgr = new Manager();

mgr.get\_info(23232, "Manager", "01/02/00");

Sales sls = new Sales();

sls.get\_info(56565, "Sales", "01/01/00");

mgr.display();

System.out.println();

sls.display();

}

}

OUTPUT:

Salary is Rs 113750.0

Details:

ID: 23232

Name: Manager

Date of joining: 01/02/00

Salary is Rs 59700.0

Details:

ID: 56565

Name: Sales

Date of joining: 01/01/00

3.

NO we cant create an abstract class.

This is evident from the following example.

abstract class abst{

abstract void m();

void display(){

System.out.println("this is an abstract class");

}

}

public class abstrct\_cant\_be{

public static void main(String[] args){

avbst c = new abst();

c.display();

}

}

OUTPUT: ERROR  
abstract\_cant\_be.java:12: error: class abstrct\_cant\_be is public, should be declared in a file named abstrct\_cant\_be.java

public class abstrct\_cant\_be{

^

abstract\_cant\_be.java:14: error: abst is abstract; cannot be instantiated

abst c = new abst();

QUESTIONS:

1. Ans.

Yes, it is necessary for an abstract class to have an abstract method and vice-versa.

2.Ans.

Yes! Abstract classes can have constructors! Yes, when we define a class to be an Abstract Class it cannot be instantiated but that does not mean an Abstract class cannot have a constructor. Each abstract class must have a concrete subclass which will implement the abstract methods of that abstract class.

3. Abs.

No, an abstract class can’t have main method.