

Q. Develop a java program to create a class with members usn, name, an array credits and array marks include methods to accept and display details (and a method to calculate SGPA of a student).

• (import "java.util.*");

→ import java.util.*;
class student {

 String usn;

 String name;

 int credits[];

 int marks[];

};

 void accept details() {

 Scanner s = new Scanner (System.in);

 System.out.println ("Enter the usn : ");

 usn = s.nextLine();

 s.nextLine();

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

 name = s.nextLine();

 s.nextLine();

 System.out.println ("Enter the no : of subjects ");

 int n = s.nextInt();

 credits = new int [n];

 marks = new int [n];

 s.nextLine();

 for (int i = 0; i < n; i++) {

 System.out.println ("Enter the credits for

 subject " (i+1) + " ");

 credits [i] = s.nextInt();

 System.out.println ("Enter the marks for subject " +

 (i+1) + " ");

Public class students especially

```
public static void main(string arg),
("student":s=new.student(),
S:acceptdetails());
```

```
display details();
System.out.println("SGPA "+calculatedSGPA());
```

10-28
10-29

100

=> Java, C, Student, Employee, Java

Java Student's gpa.
Enter the usn : 18M22CS164
Enter the Name : Anurohi
Enter the no of subjects : 5
Enter the marks : 90 85 78 92 88

Sum = Sum + credit[i] * C;

join = Sum of Credits [i,j] &g;

```
else {  
    sum = count / 2; // If n is even  
}
```

Q Develop a java program that prints all real solutions to the quadratic equation $ax^2 + bx + c = 0$ read in a, b, c and use the quadratic formula.

```
→ import java.util.Scanner;
class First {
    public static void main (String args) {
        int a, b, c;
        double r1, r2, d;
        System.out.println ("Enter the coefficient of Quadratic equation: ");
        Scanner s1 = new Scanner (System.in);
        a = s1.nextInt ();
        b = s1.nextInt ();
        c = s1.nextInt ();
        if (a == 0) {
            System.out.println ("Coefficients are invalid!");
        }
        else {
            d = b * b - (4 * a * c);
            if (d > 0) {
                System.out.println ("It has real and distinct roots");
                r1 = ((-b + Math.sqrt (d))) / (2 * a);
                r2 = ((-b - Math.sqrt (d))) / (2 * a);
                System.out.println ("The roots are " + r1 + " and " + r2);
            }
            else if (d == 0) {
                System.out.println ("It has real and equal roots");
            }
            else {
                System.out.println ("It has no real roots");
            }
        }
    }
}
```

```
→ import java.util.Scanner;
class First {
    public static void main (String args) {
        int a, b, c;
        double r1, r2, d;
        System.out.println ("Enter the coefficient of Quadratic equation: ");
        Scanner s1 = new Scanner (System.in);
        a = s1.nextInt ();
        b = s1.nextInt ();
        c = s1.nextInt ();
        if (a == 0) {
            System.out.println ("Coefficients are invalid!");
        }
        else {
            d = b * b - (4 * a * c);
            if (d > 0) {
                System.out.println ("It has real and distinct roots");
                r1 = ((-b + Math.sqrt (d))) / (2 * a);
                r2 = ((-b - Math.sqrt (d))) / (2 * a);
                System.out.println ("The roots are " + r1 + " and " + r2);
            }
            else if (d == 0) {
                System.out.println ("It has real and equal roots");
            }
            else {
                System.out.println ("It has no real roots");
            }
        }
    }
}
```

Output

```
>javac First.java
>java First
Enter the coefficients of Quadratic equation
1
2
3
It has real and equal roots
The roots are -1.0 and -1.0
```

Enter the Coefficients of Quadratic equation

```
1
1
4
It has real and distinct roots
The roots are -0.26179411928311228 and
-3.737050867568877
```

System.out.println ("The roots are " + r1 + " and " + r2);

```

4) Book Program
Public class Book {
    string name;
    double price;
    int numPages;
}

Public Book (string name, string author,
    double price, int numPages) {
    this.name = name;
    this.author = author;
    this.price = price;
    this.numPages = numPages;
}

Public void setDetails (string name, string
    author, double price, int numPages) {
    this.name = name;
    this.author = author;
    this.price = price;
    this.numPages = numPages;
}

Public string toString () {
    return "Book Details: \n Name: " + name +
    "\n Author: " + author +
    "\n Price: " + price +
    "\n Number of Pages: " + numPages;
}

Public static void main (string [] args) {
    Scanner s = new Scanner (System.in);
    int n;
    System.out.println ("enter the no. of books:");
    n = s.nextInt ();
    for (int i = 0; i < n; i++) {
        Book books [] = new Book [n];
        books [i] = new Book (s.nextLine (), s.nextLine (),
            Double.parseDouble (s.nextLine ()),
            Integer.parseInt (s.nextLine ()));
    }
}

```

Output

Enter the Number of books: 3

Book Details:

Name: Book1

Author: Author1

Price: 200.0

Number of Pages: 100

Book Details:

Name: Book2

Author: Author2

Price: 205.0

Number of Pages: 150

Book Details:

Name: Book3

Author: Author3

Price: 300.0

Number of Pages: 200

* Develop a Java program to create an abstract class named **shape** that contains two integers and an empty method named **printArea()** and three classes named **rectangle**, **triangle** and **circle** such that each one of the classes extends the class **shape**, each one of the classes contains only one method **printArea()** that prints the area of the "given" shape.

→

```

abstract class shape {
    int d1;
    int d2;
    public shape (int d1, int d2) {
        this.d1 = d1;
        this.d2 = d2;
    }
    public abstract void printArea();
}
```

class **triangle** extends **shape** {
 public void printArea() {
 double area = 0.5 * d1 * d2;
 System.out.println("Triangle Area: " + area);
 }
}

class **rectangle** extends **shape** {
 public void printArea() {
 double area = d1 * d2;
 System.out.println("Rectangle Area: " + area);
 }
}

class **circle** extends **shape** {
 public void printArea() {
 double area = Math.PI * d1 * d2;
 System.out.println("Circle Area: " + area);
 }
}

```

public void printArea() {
    double area = 0.5 * d1 * d2;
    System.out.println("Triangle Area: " + area);
}

public class shape {
    public static void main(String[] args) {
        Redangle r = new Redangle(5, 10);
        r.printArea();
        Triangle t = new Triangle(3, 6);
        t.printArea();
        Circle c = new Circle(4);
        c.printArea();
    }
}

public void printArea() {
    int area = d1 * d2;
    System.out.println("rectangle Area" + area);
}

class triangle extends shape {
    public Triangle (int base, int height) {
        Super (base, height);
    }
}

class rectangle extends shape {
    public Rectangle (int length, int width) {
        Super (length, width);
    }
}

class circle extends shape {
    public Circle (int radius) {
        Super (radius);
    }
}
```

IDE, SET package programs

```
    }  
    throw new Wrongage();  
}
```

```
student.java
package CIE;
public class student {
    String UNI;
    String name;
}
```

glass son extends father {
int fatverage, sonage;
Son(int fatverage, int sonage) throws wrongage
cannot be {

public Student {
 String name;
 int sem; }

this.name = name;
this.sem = sem;

```
    } else {
        System.out.println("father age " + Fatherage -
```

biggest numbers 1 = 1000 biggest (2, 10).

Public strong (lost strong) { 19. 3
(class: wronging even if no,)

return "age cannot be less than 0"

class cannot be extends exception

return "son cannot be older than father"

—

class Father {

Int age:

father (int age) throws wrong age
this. age = age;

catch (Wrongage e) {
 System.out.println("Exception: " + e);
}

public class student {
 public String usn;
 public String name;
 public int sem;
}

if

Enter father age :

45

Enter son age : -2
Exception: age cannot be less than 0.

public student(string usn, string name,
 int sem) {

this.usn = usn;

this.name = name;

this.sem = sem;

package cie

public class Internals {

public void calculate() {

System.out.println("Please provide

5 marks for all five courses");

int m1 = scanner.nextInt();

int m2 = scanner.nextInt();

int m3 = scanner.nextInt();

int m4 = scanner.nextInt();

int m5 = scanner.nextInt();

int sum = m1 + m2 + m3 + m4 + m5;

double avg = sum / 5.0;

System.out.println("Average marks is " + avg);

} // calculate

} // Internals

class student {

public void calculate() {

System.out.println("Please provide

5 marks for all five courses");

int m1 = scanner.nextInt();

int m2 = scanner.nextInt();

int m3 = scanner.nextInt();

int m4 = scanner.nextInt();

int m5 = scanner.nextInt();

int sum = m1 + m2 + m3 + m4 + m5;

double avg = sum / 5.0;

System.out.println("Average marks is " + avg);

} // calculate

} // student

import cie.student;

public class externals extends student {

public int calculate() {

System.out.println("Please provide

5 marks for all five courses");

int m1 = scanner.nextInt();

int m2 = scanner.nextInt();

int m3 = scanner.nextInt();

int m4 = scanner.nextInt();

int m5 = scanner.nextInt();

int sum = m1 + m2 + m3 + m4 + m5;

double avg = sum / 5.0;

System.out.println("Average marks is " + avg);

} // calculate

} // externals

```
super(username, name, sent) {  
    : = length name = 5  
}
```

Mr. Markman

else {
 System.out.println("Please provide
 marks for all courses");
}

CIE. STUDENT
CIE. INTERNALS;

class final

student

at C) shade

students[0] = new

Kernals studie

student 2 8

117 student 25

卷之三

internals student.

or $C_{int} = 0$;

System. Out. prn

system. Out.println("gemustert." + stdRandom.nextInt(5) + ".sun")

```
for (int j = 0; j < 5; j++) {  
    final marks [j] = student[
```

System. Oct. printin ("final mark")

System. Out.println();

3

卷之三

卷之三

1. 2250 4800 2000 3000 4000 5000 6000 7000 8000 9000 10000 11000 12000 13000 14000 15000 16000 17000 18000 19000 20000 21000 22000 23000 24000 25000 26000 27000 28000 29000 30000 31000 32000 33000 34000 35000 36000 37000 38000 39000 40000 41000 42000 43000 44000 45000 46000 47000 48000 49000 50000 51000 52000 53000 54000 55000 56000 57000 58000 59000 60000 61000 62000 63000 64000 65000 66000 67000 68000 69000 70000 71000 72000 73000 74000 75000 76000 77000 78000 79000 80000 81000 82000 83000 84000 85000 86000 87000 88000 89000 90000 91000 92000 93000 94000 95000 96000 97000 98000 99000 100000 101000 102000 103000 104000 105000 106000 107000 108000 109000 110000 111000 112000 113000 114000 115000 116000 117000 118000 119000 120000 121000 122000 123000 124000 125000 126000 127000 128000 129000 130000 131000 132000 133000 134000 135000 136000 137000 138000 139000 140000 141000 142000 143000 144000 145000 146000 147000 148000 149000 150000 151000 152000 153000 154000 155000 156000 157000 158000 159000 160000 161000 162000 163000 164000 165000 166000 167000 168000 169000 170000 171000 172000 173000 174000 175000 176000 177000 178000 179000 180000 181000 182000 183000 184000 185000 186000 187000 188000 189000 190000 191000 192000 193000 194000 195000 196000 197000 198000 199000 200000 201000 202000 203000 204000 205000 206000 207000 208000 209000 210000 211000 212000 213000 214000 215000 216000 217000 218000 219000 220000 221000 222000 223000 224000 225000 226000 227000 228000 229000 230000 231000 232000 233000 234000 235000 236000 237000 238000 239000 240000 241000 242000 243000 244000 245000 246000 247000 248000 249000 250000 251000 252000 253000 254000 255000 256000 257000 258000 259000 260000 261000 262000 263000 264000 265000 266000 267000 268000 269000 270000 271000 272000 273000 274000 275000 276000 277000 278000 279000 280000 281000 282000 283000 284000 285000 286000 287000 288000 289000 290000 291000 292000 293000 294000 295000 296000 297000 298000 299000 300000 301000 302000 303000 304000 305000 306000 307000 308000 309000 310000 311000 312000 313000 314000 315000 316000 317000 318000 319000 320000 321000 322000 323000 324000 325000 326000 327000 328000 329000 330000 331000 332000 333000 334000 335000 336000 337000 338000 339000 340000 341000 342000 343000 344000 345000 346000 347000 348000 349000 350000 351000 352000 353000 354000 355000 356000 357000 358000 359000 360000 361000 362000 363000 364000 365000 366000 367000 368000 369000 370000 371000 372000 373000 374000 375000 376000 377000 378000 379000 380000 381000 382000 383000 384000 385000 386000 387000 388000 389000 390000 391000 392000 393000 394000 395000 396000 397000 398000 399000 400000 401000 402000 403000 404000 405000 406000 407000 408000 409000 410000 411000 412000 413000 414000 415000 416000 417000 418000 419000 420000 421000 422000 423000 424000 425000 426000 427000 428000 429000 430000 431000 432000 433000 434000 435000 436000 437000 438000 439000 440000 441000 442000 443000 444000 445000 446000 447000 448000 449000 450000 451000 452000 453000 454000 455000 456000 457000 458000 459000 460000 461000 462000 463000 464000 465000 466000 467000 468000 469000 470000 471000 472000 473000 474000 475000 476000 477000 478000 479000 480000 481000 482000 483000 484000 485000 486000 487000 488000 489000 490000 491000 492000 493000 494000 495000 496000 497000 498000 499000 500000 501000 502000 503000 504000 505000 506000 507000 508000 509000 510000 511000 512000 513000 514000 515000 516000 517000 518000 519000 520000 521000 522000 523000 524000 525000 526000 527000 528000 529000 530000 531000 532000 533000 534000 535000 536000 537000 538000 539000 540000 541000 542000 543000 544000 545000 546000 547000 548000 549000 550000 551000 552000 553000 554000 555000 556000 557000 558000 559000 560000 561000 562000 563000 564000 565000 566000 567000 568000 569000 570000 571000 572000 573000 574000 575000 576000 577000 578000 579000 580000 581000 582000 583000 584000 585000 586000 587000 588000 589000 589000 590000 591000 592000 593000 594000 595000 596000 597000 598000 599000 600000 601000 602000 603000 604000 605000 606000 607000 608000 609000 610000 611000 612000 613000 614000 615000 616000 617000 618000 619000 620000 621000 622000 623000 624000 625000 626000 627000 628000 629000 630000 631000 632000 633000 634000 635000 636000 637000 638000 639000 640000 641000 642000 643000 644000 645000 646000 647000 648000 649000 650000 651000 652000 653000 654000 655000 656000 657000 658000 659000 660000 661000 662000 663000 664000 665000 666000 667000 668000 669000 670000 671000 672000 673000 674000 675000 676000 677000 678000 679000 680000 681000 682000 683000 684000 685000 686000 687000 688000 689000 689000 690000 691000 692000 693000 694000 695000 696000 697000 698000 699000 700000 701000 702000 703000 704000 705000 706000 707000 708000 709000 710000 711000 712000 713000 714000 715000 716000 717000 718000 719000 720000 721000 722000 723000 724000 725000 726000 727000 728000 729000 729000 730000 731000 732000 733000 734000 735000 736000 737000 738000 739000 739000 740000 741000 742000 743000 744000 745000 746000 747000 748000 749000 749000 750000 751000 752000 753000 754000 755000 756000 757000 758000 759000 759000 760000 761000 762000 763000 764000 765000 766000 767000 768000 769000 769000 770000 771000 772000 773000 774000 775000 776000 777000 778000 779000 779000 780000 781000 782000 783000 784000 785000 786000 787000 788000 789000 789000 790000 791000 792000 793000 794000 795000 796000 797000 798000 799000 799000 800000 801000 802000 803000 804000 805000 806000 807000 808000 809000 809000 810000 811000 812000 813000 814000 815000 816000 817000 818000 819000 819000 820000 821000 822000 823000 824000 825000 826000 827000 828000 829000 829000 830000 831000 832000 833000 834000 835000 836000 837000 838000 839000 839000 840000 841000 842000 843000 844000 845000 846000 847000 848000 849000 849000 850000 851000 852000 853000 854000 855000 856000 857000 858000 859000 859000 860000 861000 862000 863000 864000 865000 866000 867000 868000 869000 869000 870000 871000 872000 873000 874000 875000 876000 877000 878000 879000 879000 880000 881000 882000 883000 884000 885000 886000 887000 888000 889000 889000 890000 891000 892000 893000 894000 895000 896000 897000 898000 898000 899000 900000 901000 902000 903000 904000 905000 906000 907000 908000 909000 909000 910000 911000 912000 913000 914000 915000 916000 917000 918000 919000 919000 920000 921000 922000 923000 924000 925000 926000 927000 928000 929000 929000 930000 931000 932000 933000 934000 935000 936000 937000 938000 939000 939000 940000 941000 942000 943000 944000 945000 946000 947000 948000 949000 949000 950000 951000 952000 953000 954000 955000 956000 957000 958000 959000 959000 960000 961000 962000 963000 964000 965000 966000 967000 968000 969000 969000 970000 971000 972000 973000 974000 975000 976000 977000 978000 978000 979000 980000 981000 982000 983000 984000 985000 986000 987000 988000 989000 989000 990000 991000 992000 993000 994000 995000 996000 997000 998000 998000 999000 999000 1000000

卷之三

卷之三

卷之三

112

卷之三

return d;

E return type 4 () {
 return e;

}

class Generic <A,B,C,D,E> {
 A a;
 B b;
 C c;
 D d;
 E e;

 Generic (A a1, B b1, C c1, D d1, E e1)
 a = a1;
 b = b1;
 c = c1;
 d = d1;
 e = e1;

 }
}

Void show type () {
 System.out.println (a. getClass (). getName());
 (b. getClass (). getName());
 (c. getClass (). getName());
 (d. getClass (). getName());
 (e. getClass (). getName());
}

A return type () {
 return a;

B return type 1 () {
 return b;

}

C return type 2 () {
 return c;

}

D return type 3 () {
 return d;

}

E return type 4 () {
 return e;

}

class Generic demo {
 public static void main (String[] args) {
 Generic <Integer, float, String, Double, Boolean>
 g1 = new Generic (Integer, float, String, Double, Boolean);
 g1. showtype ();
 System.out.println
 int x = g1. return type ();
 System.out.println (x);
 float x1 = g1. return type ();
 System.out.println (x1);
 String x2 = g1. return type ();
 System.out.println (x2);
 Double x3 = g1. return type 3 ();
 System.out.println (x3);
 Boolean x4 = g1. return type 4 ();
 System.out.println (x4);
 }

Java. lang. Integer
Java. lang. float
Java. lang. String
Java. lang. double
Java. lang. Boolean

1
1. 23222. 11332
true.

BMS

Q) write a program to show user-defined exception.
 Create class employee as inheritance. Show exception when manager salary is less than employee.

```
class Manager extends Exception {
    public String toString() {
        return "Manager salary Cannot be less than worker".
```

```
class Manager {
    int managerSalary;
```

```
public Manager(int managerSalary) {
    this.managerSalary = managerSalary;
```

```
} class Worker extends Manager {
    int workerSalary;
```

```
public Worker(int workerSalary, int managerSalary)
    throws MyException {
    if (managerSalary < workerSalary) {
        throw new MyException();
    }
}
```

```
public class Employee {
    public static void main(String[] args) {
        Worker w = new Worker(40000, 30000);
        System.out.println(w);
```

```
public class Manager {
    int managerSalary;
```

```
public Manager(int managerSalary) {
    this.managerSalary = managerSalary;
```

```
int workerSalary;
```

```
public Worker(int workerSalary) {
    this.workerSalary = workerSalary;
```

g) Write a program which creates: Two semesters
including "One college of Engineering" once
and another displacement.

=> class display Thread extends Thread {
 public void run() {
 System.out.println("display message");
 }
}

```
private void  
private int interval;  
private int interval; (String message  
public DisplayThread (String message  
public int interval);
```

JUN 2015

This message = message!

This interval is called an interval.

```
public void run()
```

white (true) }

Original
Superior print line (message);

bread sleep (interval)

Each interrupted exception class

for wind-up print back trace (v)

1. 1 2. 2 3. 3 4. 4 5. 5 6. 6 7. 7 8. 8 9. 9 10. 10

class Display & message { } needs void &

public static void main (String [] args) {

bread bus meat = new display meat

Thread & use Thread - new Display Thread

develop a Java program to create a class that maintains two kind of accounts. Bank that maintains savings account for customers, one called current account and the other current account. the savings account provides compound interest and account provides basic no checkbook facility withdrawal facility cheque book facility. The current account provides holders should also no interest current account and if the balance falls below this level a service charge is imposed. Create a class account that stores data customer name, account number and type of account from this derive the classes current account and savings account to make them more specific. their requirements include the necessary methods in order to achieve the following tasks.

- Accept deposit from customer & update the balance
- display the balance
- compute & deposit interest
- permit withdraw & update the balance. check for minimum balance impose penalty & update the balance.

```

import java.util.Scanner;
class Bank {
    double accno;
    String name;
    String type;
    double balance;
    public void display() {
        System.out.println("Name: " + name);
        System.out.println("Type: " + type);
        System.out.println("Balance: " + balance);
    }
    public void deposit(double deposit) {
        this.balance += deposit;
    }
    public void withdraw(double amount) {
        if (amount > balance) {
            System.out.println("Insufficient balance");
        } else {
            balance -= amount;
            System.out.println("After withdrawal, the
balance is: " + balance);
        }
    }
}

```

class savings extends Bank {
 double rate;
 int time;
}

public savings (double acc no, string name,
 double balance, int time, double rate) {
 super (acc no, name, "savings", balance);
 this.time = time; // always starts with 0
 this.rate = rate; // always

}

public void calculateInterest () {
 balance += (balance * time * rate) / 100;
}

class current extends Bank {
 double balance, minBalance;
}

public current (double acc no, string name, double
 balance, double minBalance) {
 super (acc no, name, "current", balance);
 this.minBalance = minBalance;

}

public void applyServiceCharge () {
 if (balance < minBalance) {
 System.out.println ("service charge of 5% is
 applied");
 balance -= balance * 0.05;
 }
}

if (balance < minBalance) {
 System.out.println ("service charge of 5% is applied");
 balance -= balance * 0.05;

254

```
9. DrawString (out, 100, 200);  
  
public static void main (String args [ ]) {  
    Division dm = new Division (main ());  
    dm. setSize (new Division (200, 400));  
    dm. setTitle ("Division of integers");  
    dm. setVisible (true);  
}
```

of $\frac{1}{2} \sin 120^\circ = \frac{1}{2} \times \frac{\sqrt{3}}{2} = \frac{\sqrt{3}}{4}$

Result Result: 12 0 infinity
(a) out of 3 solutions (abs)

Number 1: 12 Number 2: a

Number of errors except on 1.0

Java: Long, Number Format, Exception, for
many situations!

Number 1: 10 Number 2: 12

Result Result 0 0 1/2

Result Result 12.05.000

19. October 1910. bion. siting.

2