**1.Find the maximum of three numbers**

Code:

public class Maximum

{

public static void main(String args[])

{

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

int y=Integer.parseInt(args[1]);

int z=Integer.parseInt(args[2]);

System.out.println("Max: "+calMax(x,y,z));

}

static int calMax(int x,int y,int z)

{

int max=0;

if(x>y)

{ if(x>z)

max=x;

else

max=z;

}

else

{ if(y>z)

max=y;

else

max=z;

}

return max;

}

}

**2 .Write a program to check whether the input alphabet is a vowel or not.**

Code:

public class Vowel

{

public static void main(String args[])

{

String c=args[0].toLowerCase();

char alph=c.charAt(0);

checkAlpha(alph);

}

static void checkAlpha(char alph)

{

switch(alph)

{ case 'a':

case 'e':

case 'i':

case 'o':

case 'u':System.out.println("vowel!!"); break;

default:System.out.println("Consonant!!");

}

}

}

**3.Develop a program, that accepts a deposit amount and calculates amount of interest the deposit amount earns in an year. The bank pays a flat 4% for deposits of up to Rs.1000, a flat 4.5% per year for deposits of up to Rs.5000, and a flat 5% for deposits of more than Rs.5000.**

Code:

public class Interest

{

public static void main(String args[])

{

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

System.out.print("Interest: " + calInterest(amt));

}

static double calInterest(int amt)

{

double r=0;

if (amt<=1000)

r=0.04;

if (amt>1000 && amt<=5000)

r=0.045;

if(amt>5000)

r=0.05;

double interest=r\*amt;

return interest;

}

}

**4.Develop the program, which accepts the gross pay and produces the amount of tax owed. For a gross pay of $240 or less, the tax is 0%; for over $. 240 and $. 480 or less, the tax rate is 15%; and for any pay over $480, the tax rate is 28%.**

Code:

public class GrossPay

{

public static void main(String args[])

{

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

System.out.print("Tax Owed: " + calTax(gp));

}

static double calTax(int gp)

{

double tax=0;

if (gp<=240)

tax=0;

if (gp>240 && gp<=480)

tax=0.15;

if (gp>480)

tax=0.28;

double owedTax=tax\*gp;

return owedTax;

}

}

**5.Some credit card companies pay back a small portion of the charges a customer makes over a year. A particular credit card company's pay back policy is as follows:  
1.0.25% for charges up to Rs. 500.  
2.0.50% for the next Rs.1000 (that is, the portion between Rs. 500 and Rs. 1500),  
3.0.75% for the next Rs.1000 (that is, the portion between Rs. 1500 and Rs. 2500),  
4.1.0% for charges above Rs2500.  
Thus, a customer who charges Rs. 400 a year receives Rs.1.00, which is 0.25 · 1/100 · 400, and one who charges Rs1, 400 a year receives Rs. 5.75, which is 1.25 = 0.25 · 1/100 · 500 for the first Rs. 500 and 0.50 · 1/100 · 900 = 4.50 for the next Rs. 900. Determine by hand the pay backs amount for a customer who charged Rs. 2000 and one who charged Rs. 2600.  
Define the program, which accepts a charge amount and computes the corresponding pay back amount.**

Code:

public class Charge

{

public static void main(String args[])

{

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

System.out.print("Pay Back amount: " +calculatePay(charge));

}

static double calculatePay(int charge)

{

double pay=0;

if (charge<=500)

pay=0.0025\*charge;

if (charge>500 && charge<=1500)

pay=0.0025\*500+0.0050\*(charge-500);

if (charge>1500 && charge<=2500)

pay=0.0025\*500+0.0050\*1000+0.0075\*(charge-1500);

if (charge>2500)

pay=0.0025\*500+0.0050\*1000+0.0075\*1000+0.01\*(charge-2500);

return pay;

}

}

**6.Implement a method that returns the day of the week for a given day (1..31), month (1..12) and year.  
The day of the week of dates between March 1900 and February 2100 can be calculated as follows:  
1.Calculate the total number of days from 1900/1/1 to the given date (see below, for details).  
2.Divide this number by 7, the remainder is the day of the week (0 - sunday, 1 - monday, etc)  
To calculate the total number of days:  
1.Subtract 1900 from the given year and multiply the result by 365. Add the missing leaps years by adding (year - 1900) / 4.  
2.If the given year is a leap year and the month is January or February, subtract 1 from the previous result.  
3.Add no of days of the given year till the given month (in case of February always add 28, because the additional day for a leap year would have been already added).**

Code:

public class Days

{

public static void main(String args[])

{

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

int m=Integer.parseInt(args[1]);

int y=Integer.parseInt(args[2]);

int flag=checkLeapYear(y);

int x=y-1900;

x=x\*365;

x=x+((y-1900)/4);

if ((flag==1)&&(m==1 || m==2))

x=x-1;

x=x+noOfDays(m,d);

dayToDay(x%7);

}

static int checkLeapYear(int y)

{

int flag=0;

if(y%100==0)

{ if(y%400==0)

flag=1;

}

else

{ if(y%4==0)

flag=1;

}

return flag;

}

static void dayToDay(int rem)

{

switch(rem)

{ case 0:System.out.print("Sunday");

break;

case 1:System.out.print("Monday");

break;

case 2:System.out.print("Tuesday");

break;

case 3:System.out.print("Wednesday");

break;

case 4:System.out.print("Thursday");

break;

case 5:System.out.print("Friday");

break;

case 6:System.out.print("Saturday");

break;

}

}

static int noOfDays(int m,int d)

{ int days=0;

switch(m)

{ case 1:days=d;

break;

case 2:days=31+d;

break;

case 3:days=31+28+d;

break;

case 4:days=31+28+31+d;

break;

case 5:days=31+28+31+30+d;

break;

case 6:days=31+28+31+30+31+d;

break;

case 7:days=31+28+31+30+31+30+d;

break;

case 8:days=31+28+31+30+31+30+31+d;

break;

case 9:days=31+28+31+30+31+30+31+31+d;

break;

case 10:days=31+28+31+30+31+30+31+31+30+d;

break;

case 11:days=31+28+31+30+31+30+31+31+30+31+d;

break;

case 12:days=31+28+31+30+31+30+31+31+30+31+30+d;

}

return days;

}

}

**7. Write a program to automate the following loan policy.**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Age category | Gender | Profession | Personal assets | Loan amount eligible |
| 16 -25 | M /F | Self-Employed | >25000 | 10000 |
| Professional | 15000 |  |  |  |
| 26 - 40 | M | SelfEmployed / Professional | > 40000 | 25000 |
| F | 30000 |  |  |  |
| 41 - 60 | M / F | SelfEmployed / Professional | > 50000 | 40000 |
| > 60 | M/F | Self Employed | > 25000 | 35000 – Age \* 100 |
| Retired | 25000 – Age \* 100 |  |  |  |

**Write a program that accepts age, gender, job status and assets, and return the eligible loan amount.**

Code:

public class Loan

{

public static void main(String args[])

{

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

String c=args[1].toUpperCase();

char gender=c.charAt(0);

String job=args[2];

int asset=Integer.parseInt(args[3]);

System.out.println("loan amount: " + calLoan(age,gender,job,asset));

}

static int calLoan(int age,char gender,String job,int asset)

{

int loan=0;

if ((age>=16)&&(age<=25))

{ if(gender=='M' || gender=='F')

if(asset>25000)

{ if(job.equals("selfemployed"))

loan=10000;

else

loan=15000;

}

}

if((age>=26)&&(age<=40))

{ if ((job.equals("selfemployed"))||(job.equals("professional")))

if(asset>40000)

{ if(gender=='M')

loan=25000;

else

loan=30000;

}

}

if((age>=41)&&(age<=60))

{ if ((job.equals("selfemployed"))||(job.equals("professional")))

if(gender=='M' || gender=='F')

if(asset>40000)

loan=40000;

}

if(age>60)

{ if(gender=='M' || gender=='F')

if(asset>25000)

if(job.equals("selfemployed"))

loan=35000-age\*100;

else

loan=25000-age\*100;

}

return loan;

}

}