16-09-16

***Given two non-negative int values, return true if they have the same last digit, such as with 27 and 57. Note that the % "mod" operator computes remainders, so 17 % 10 is 7. The sameLast() method should return a boolean value***

***Sample:***

***sameLast(25, 35) ---> “true”***

***sameLast(21, 39) ---> “false”***

***Note: Create a Solution1 class which contains a single method called sameLast(int, int). Create another class TestSolution1 which contains the main() method, instantiates Solution1 object and tests it with above values.***

public class solution1

{

public boolean sameLast(int a,int b)

{

int c,d;

c=a%10;

d=b%10;

if(c==d)

return true;

else

return false;

}

}

import java.util.\*;

public class TestSolution1

{

public static void main(String args[])

{

solution1 s=new solution1();

Scanner sc=new Scanner(System.in);

int a=sc.nextInt();

int b=sc.nextInt();

boolean bln=s.sameLast(a,b);

System.out.println(bln);

}

}

***Given 2 int values, return true if either of them is in the range 10..20 inclusive.***

***Sample:***

***inRange(20, 19) ---> “true”***

***inRange(30, 100) ---> “false”***

import java.util.\*;

public class solution2

{

public static boolean inRange(int a,int b)

{

if((a<=20)&&(a>=10)&&(b<=20)&&(b>=10))

return true;

else

return false;

}

public static void main(String args[])

{

Scanner sc=new Scanner(System.in);

int a=sc.nextInt();

int b=sc.nextInt();

System.out.println(inRange(a,b));

}

}

***Find sum all the integers between a given range(inclusive), that are divisible by 13, 15 or 17, but not by 30***

***Sample:***

***sumInt(1, 30) 71***

***sumInt(10, 15) 28***

import java.util.\*;

public class solution3

{

public static void main(String args[]){

Scanner sc=new Scanner(System.in);

int a=sc.nextInt();

int b=sc.nextInt();

System.out.println(sumInt(a,b));

}

public static int sumInt(int a,int b){

int sum = 0;

// from 1 to 1000

for (int num = a; num <= b; num++) {

if (num % 30 == 0) {

continue; // If it's divisible by 30, skip it.

}

if (num % 13 == 0 || num % 15 == 0 || num % 17 == 0) {

sum += num; // If it's divisible by 13 or 15 or 17 add it to sum.

}

}

return(sum);

}

}

***Find the number of leap years in a given range(inclusive)***

***countLeapYear(2000, 2016) 5***

***Note: A leap should be divisible by 4, should not be divisible by 100, but can be divisible by 400. Leap years : 4, 16, 96, 400, 800, 2000***

***Non-Leap years : 100, 200, 300, 500 ..***

import java.util.\*;

public class solution4{

public static int countLeapYear(int a,int b)

{

int sum=1,num;

for(num=a;num<=b;num++){

if((num%4==0||num%400==0)&&(num%100!=0))

sum++;

}

return sum;

}

public static void main(String args[])

{

Scanner sc=new Scanner(System.in);

int a=sc.nextInt();

int b=sc.nextInt();

System.out.println(countLeapYear(a,b));

}

}