**CSE1007 – JAVA PROGRAMMING**

Lab Exercise on Methods

**Question 1**

Write a JAVA program to find the GCD of any two numbers. Your program should have method findGCD() that return the gcd of the two numbers.

**CODE:**

import java.util.Scanner;

public class q1{

public static int findGCD(int a, int b)

{

int i,min,GCD=1;

min = (a < b) ? a : b;

for(i=2;i<=min;i++)

{

if(a%i==0 && b%i==0)

GCD=i;

}

return GCD;

}

public static void main(String args[])

{

Scanner in = new Scanner(System.in);

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

int n1=in.nextInt();

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

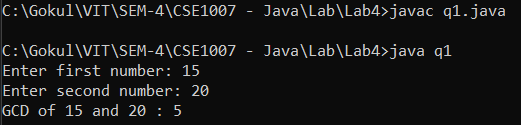
int n2=in.nextInt();

System.out.println("GCD of "+n1+" and "+n2+" : "+findGCD(n1,n2));

}

}

**OUTPUT:**

****

**Question 2**

Travel Tickets Company sells tickets for airlines, tours, and other travel-related services. Because long numbers have often been entered incorrectly by agents, Travel Tickets has asked you to code a JAVA program that will indicate if a ticket number entry is invalid. Ticket numbers are 11 digits long. Ticket numbers are designed such that if you drop the last digit of the number, then divide the 10-digit number by 7, the remainder of the division will be identical to the last dropped digit. If ticket number is 10-digits, include the 11th digit or if it is 11-digit long, check for the validity. If the ticket number is any other length, your program should prompt the agent to check and re-enter the ticket number. Include a method **isValidTicket()** that return either true or false depending on the validity of the ticket.

**CODE:**

import java.util.Scanner;

public class q2{

public static boolean isValidTicket(long tID)

{

int flag=0;

String str = String.valueOf(tID);

if(str.length()==11)

{

long lastDigit = tID%10;

long n = tID/10;

if(lastDigit==n%7)

flag=1;

}

if(flag==1)

return true;

else

return false;

}

public static void main(String args[])

{

Scanner in = new Scanner(System.in);

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

long n=in.nextLong();

String s = String.valueOf(n);

if(s.length()==10)

{

n=(n\*10)+(n%7);

System.out.println("11 Digit Ticket number: "+n);

}

if(isValidTicket(n))

System.out.println("Ticket number is valid");

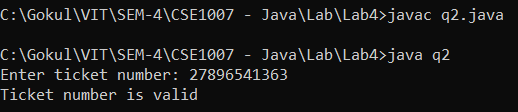
else

System.out.println("Kindly check and renter the ticket number");

}

}

**OUTPUT:**



**Question 3**

Assume that you have a list of words and you wish to find how many words are palindrome in the list. Devise a JAVA program that reads several words and displays the palindrome words and count of such words. Your program should have two methods namely

**String** **reverseString(String)** and **boolean** **isPalindrome(String)**.

Hint: A word is palindrome if its reverse is same as the original.

**CODE:**

import java.util.Scanner;

public class q3{

public static String reverseString(String str)

{

String rev="";

char ch;

for (int i=str.length()-1; i>=0; i--)

{

ch= str.charAt(i);

rev+=ch;

}

return rev;

}

public static boolean isPalindrome(String s){

if(s.equals((reverseString(s))))

return true;

else

return false;

}

public static void main(String args[])

{

Scanner in = new Scanner(System.in);

int n,i,c=0;

System.out.print("Enter no of words: ");

n=in.nextInt();

String words[]=new String[n];

System.out.print("Enter "+n+" words: ");

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

words[i]=in.next();

System.out.println();

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

{

if(isPalindrome(words[i]))

{

c++;

System.out.println(words[i]+" is Palindrome (Reverse: "+reverseString(words[i])+")");

}

}

if(c>0)

System.out.println("\nNo of plaindrome words: "+c);

else

System.out.println("\nNo plaindrome words found");

}

}

**OUTPUT:**

