King Saud University

Collage of Computer and Information Science

Department of Computer Science

Discrete Mathematics for Computer Science CSC 281

Fall 2017

**PROJECT (ISBN)**

**By**:

Tala Ben Dlim

436100857

Abdulwahab Alboriydi

436106748

Ahmad Alabandi

436101544

**Instructor:**

Dr. Aqil Mohammed

# **Table of Contents**

[Table of Contents 1](#_Toc501984632)

[1. Description 2](#_Toc501984633)

[2. Data Structure Used 2](#_Toc501984634)

[3. Sample Run: 2](#_Toc501984635)

[7. Conclusion 4](#_Toc501984636)

1. Description:

In this project we started by designing an ISBN validator that only checks if the ISBN is valid or not. The second approach was to think of an algorithm to solve for the missing digit. Since the options are only 10 (11 if the missing number is the last) so we decided to let the program try them all and see what fits the formula if non found it is then not solvable. Here an example:

15688?111X

So it should be: 10\*1 + 9\*5 + 8\*6 + 7\*8 + 6\*8 + 5\*? + 4\*1 + 3\*1 + 2\*1 + 1\*10 mod 11 = 0

= 226 +5 \*? mod 11 =0

Then we try from 0 up to 9 (we stop when we find the solution.

0: 226 + (5\*0) mod 11 = 226 mod 11 = 6 - Not correct.

1: 226 + (5\*1) mod 11 = 231 mod 11 = 0 - Correct

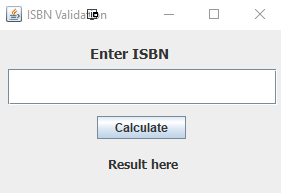
So the answer is 1 and the ISBN = 156881111X

2. Data Structure Used:

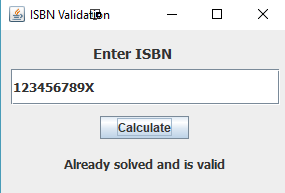
The project implemented using Java. We used Array of char of size 10 each character represent a digit. The data wasn’t large so the choices depend on the easier to implement and more organized. Also Arrays are indexed so it is easier to know the position of the digit by the index number.

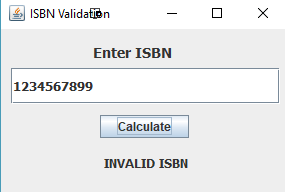
# 3. Sample Run:

User Interface:

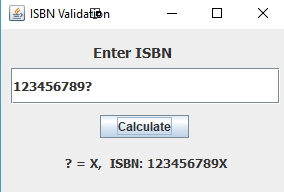


Only validating:

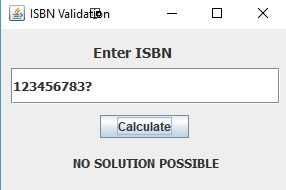




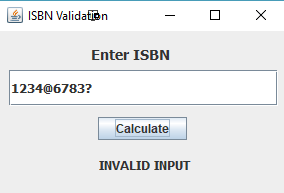
Solving:



No Solution:



Invalid input:



7. Conclusion:

In conclusion, the idea of the project was to solve the missing digit of an ISBN and check if there is a solution. We went through the steps of the project starting from the GUI until the final result. We had some difficulties at the beginning but after some testing and debugging all of them were solved.

Source Code:

import java.awt.EventQueue;

import javax.swing.JFrame;

import javax.swing.JTextField;

import java.awt.BorderLayout;

import javax.swing.JLabel;

import java.awt.Font;

import javax.swing.JButton;

import javax.swing.SwingConstants;

import java.awt.event.ActionListener;

import java.awt.event.ActionEvent;

public class GUIisbn {

private JFrame frmIsbnValidation;

private JTextField ISBNtf;

/\*\*

\* Launch the application.

\*/

public static void main(String[] args) {

EventQueue.invokeLater(new Runnable() {

public void run() {

try {

GUIisbn window = new GUIisbn();

window.frmIsbnValidation.setVisible(true);

} catch (Exception e) {

e.printStackTrace();

}

}

});

}

/\*\*

\* Create the application.

\*/

public GUIisbn() {

initialize();

}

/\*\*

\* Initialize the contents of the frame.

\*/

private void initialize() {

frmIsbnValidation = new JFrame();

frmIsbnValidation.setTitle("ISBN Validation");

frmIsbnValidation.setBounds(100, 100, 301, 202);

frmIsbnValidation.setDefaultCloseOperation(JFrame.EXIT\_ON\_CLOSE);

frmIsbnValidation.getContentPane().setLayout(null);

ISBNtf = new JTextField();

ISBNtf.setFont(new Font("Tahoma", Font.BOLD, 13));

ISBNtf.setBounds(10, 39, 269, 36);

frmIsbnValidation.getContentPane().add(ISBNtf);

ISBNtf.setColumns(10);

JLabel lblEnterIsbn = new JLabel("Enter ISBN");

lblEnterIsbn.setHorizontalAlignment(SwingConstants.CENTER);

lblEnterIsbn.setFont(new Font("Tahoma", Font.BOLD, 14));

lblEnterIsbn.setBounds(85, 11, 92, 25);

frmIsbnValidation.getContentPane().add(lblEnterIsbn);

JLabel resultf = new JLabel("Result here");

resultf.setHorizontalAlignment(SwingConstants.CENTER);

resultf.setFont(new Font("Tahoma", Font.BOLD, 12));

resultf.setBounds(56, 116, 178, 36);

JButton btnCalculate = new JButton("Calculate");

btnCalculate.addActionListener(new ActionListener() {

public void actionPerformed(ActionEvent e) {

String st =ISBNtf.getText();

if(st.length()!=10)

{

resultf.setText("ISBN Length must be 10");

return;

}

ISBNValidation isbn = new ISBNValidation(st);

char q = isbn.findQ();

if(!isbn.isValid())

{

resultf.setText("INVALID INPUT");

}

else if(q=='?')

{

resultf.setText("NO SOLUTION POSSIBLE");

}

else if(q=='s')

{

if( isbn.validate())

resultf.setText("Already solved and is valid ");

else

resultf.setText("INVALID ISBN");

}

else

{

isbn.solve();

resultf.setText("? = "+q+", ISBN: "+isbn.toString());

}

}

});

btnCalculate.setBounds(99, 86, 89, 23);

frmIsbnValidation.getContentPane().add(btnCalculate);

frmIsbnValidation.getContentPane().add(resultf);

}

}

**public** **class** **ISBNValidation** {

**char**[] ISBN;

**public** **ISBNValidation**(){

ISBN = **new** **char**[10];

**for**(**int** **i**=0;i<10;i++)

{

ISBN[i]='0';

}

}

**public** **ISBNValidation**(**String** s){

ISBN = **new** **char**[10];

**for**(**int** **i**=0;i<10;i++)

{

ISBN[i]=s.charAt(i);

}

}

**public** **boolean** **isValid**()

{

**boolean** **x**=**false**,**q**=**false**;

**for**(**int** **i**=0;i<10;i++)

{

**if**(ISBN[i]=='X'&&!x)

x=**true**;

**else** **if**(ISBN[i]=='?'&&!q)

q=**true**;

**else** **if**(!**Character**.*isDigit*(ISBN[i]))

**return** **false**;

}

**return** **true**;

}

**public** **boolean** **isValid**(**String** s)

{

**boolean** **x**=**false**,**q**=**false**;

**for**(**int** **i**=0;i<10;i++)

{

**if**(s.charAt(i)=='X'&&!x)

x=**true**;

**else** **if**(s.charAt(i)=='?'&&!q)

q=**true**;

**else** **if**(!**Character**.*isDigit*(s.charAt(i)))

**return** **false**;

}

**return** **true**;

}

**public** **void** **solve**(){

**char** **q** = findQ();

**for**(**int** **i**=0;i<10;i++)

**if**(ISBN[i]=='?')

ISBN[i]=q;

}

**public** **boolean** **isSolved**(){

**if**(!isValid())

**return** **false**;

**for**(**int** **i**=0;i<10;i++)

**if**(ISBN[i]=='?')

**return** **false**;

**return** **true**;

}

**public** **boolean** **validate**()

{

**if**(!isSolved() || !isValid())

**return** **false**;

**int** **sum**=0;

**for**(**int** **i**=0;i<10;i++){

**if**(ISBN[i]!='?'&&ISBN[i]!='X'){

sum+=**Character**.*getNumericValue*(ISBN[i])\*(10-i);

}

**else**

sum+=10\*(10-i);

}

**if**(sum%11==0)

**return** **true**;

**return** **false**;

}

**public** **char** **findQ**(){

**if**(isSolved())

**return** 's';

**if**(!isValid())

**return** 'n';

**char** **q**='?';

**int** **sum**=0;

**int** **qindex**=0;

**for**(**int** **i**=0;i<10;i++){

**if**(ISBN[i]!='?'&&ISBN[i]!='X'){

sum+=**Character**.*getNumericValue*(ISBN[i])\*(10-i);

}

**else** **if**(ISBN[i]=='X'){

sum+=10\*(10-i);

}

**else**{

qindex=10-i;

}

}

**int** **isx** =10;

**if**(qindex==1)

isx=11;

**for**(**int** **i**=1;i<isx;i++)

{

**if**((((qindex\*i)+sum)%11==0))

{

**if**(i!=10){

q=(**char**)(i+48);

}

**else**

q='X';

}

}

**return** q;

}

**public** **String** **toString**()

{

**String** **s** = "";

**for**(**int** **i**=0;i<10;i++)

{

s+=ISBN[i];

}

**return** s;

}

}