Week 4 Project—Gotos and Regular Expressions

**Scenario**

In this week’s project you will look at Designing code and regular expressions

**Rubric**

Point distribution for this activity:

|  |  |  |
| --- | --- | --- |
| **Activity** | | |
| **Document** | **Points possible** | **Points received** |
| Part A | 20 |  |
| Part B | 30 |  |
| **Total Points** | **50** |  |

Part A:

In a letter to the editor of CACM, Rubin (1987) uses the following code segment as evidence that the readability of some code with gotos is better than the equivalent code without gotos. This code finds the first row of an n by n integer matrix named x that has nothing but 0 values.

for (i = 1; i <= n; i++) {

for (j = 1; j <= n; j++)

if (x[i][j] != 0)

goto reject;

println ('First ­all-​­zero row is:', i);

break;

reject:

}

Rewrite this code without gotos in any language of your choice. Compare the readability of your code to that of the example code (should be at least 1 paragraph).

for (int i = 1; i<= n; i++){

innerBreak = false;

for (int j = 1; j <= n; j++){

if(x[i][j] != 0){

innerBreak = true;

break;

}

}

If(!innerBreak){

System.out.println("First -all--zero row is" + i);

}

break;

}

Seeing as the goto statement simply moves to the next instance of the outer for loop, a continue statement there would only move forward the inner for loop, so breaking out the inner loop and moving the outer loop to the next instance would be the best way out. The unfortunate part here is to ensure that the print will not go out if the inner loop is broken, so a bool value stating if the inner loop is broken is put in, allowing the print to only happen if the inner is not broken during the whole loop. Readability suffers greatly due to this, and while this is something that I really threw together quickly, it is generally how I would handle something like this in my own experience. Because of this, I can fully understand where the code is going, and how the bool affects the secondary process of the loops, but that does not mean that everyone who looks at it will understand the same.

Part B:

Research online about pattern matching and regular expressions.

1. What is a regular expression? A pattern-recognition system that uses expressions like \d{3} to signify a specific formatting of a string.
2. Locate an example online and explain what it does: /[\w.\_%+=-][+@[\w.-]+\.[a-zA-Z]{2-4}/](mailto:+@[\w.-%5d+\.%5ba-zA-Z%5d%7b2-4%7d/) - this looks for email addresses. \w is any character a-z upper or lower, /\_%=+- adds those specific characters to the search, and + signifies that the search can return any number of items, followed by a match @ exactly, with any number of characters a-z upper or lower including .-, followed by a . and then matching upper or lower a-z for at least 2 up to 4 times.
3. We are going to use the following regular expression to search for a phone number: \(?\d{3}\)?[. -]? \*\d{3}[. -]? \*[. -]?\d{4}
   1. What does the \d indicate? - Digit or integer
   2. What does the {3} indicate? – the amount of the previous signifier, in this case it would be 3 digits
4. Feel free to create a form/GUI or use console to create a regular expression to determine if a phone number input is valid. You will need to import a regular expression library. My code below as an example:

string phoneNum =txtName.Text;

string MatchPhonePattern = @"\(?\d{3}\)?[. -]? \*\d{3}[. -]? \*[. -]?\d{4}";

Regex rx = new Regex(MatchPhonePattern, RegexOptions.Compiled | RegexOptions.IgnoreCase);

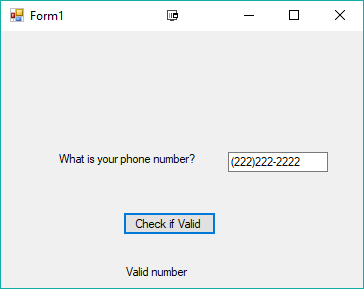
Match match = rx.Match(phoneNum);

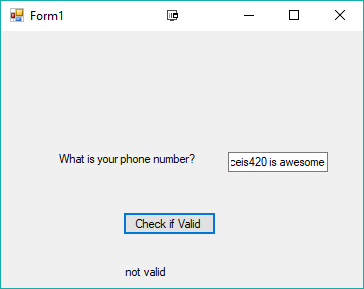
if (match.Success)

lblDisplay.Text = "Valid number";

else

lblDisplay.Text = "not valid";





Run your program and include a screenshot of your code running and copy and paste your code. Use phone number as a regular expression, then validate a social security number using regular expressions.

Code for Phone number regex:

private void jButtonMouseClicked(java.awt.event.MouseEvent evt){

If (findPhoneNumber(txtInputPhoneNumber.getText())){  
 lblOutputText.setText(“Valid phone number”);

}

else{

lblOutputText.setText(“Not valid phone number”);

}

}

private Boolean findPhoneNumber(String checkThis){

Pattern phoneNumber = Pattern.compile(“\\(?\\d{3}\\)?[. -]? \*\\d{3}[. -]?\\d{4}”);

Matcher checkingPossible = phoneNumber.matcher(checkThis);

return checkingPossible.find();

}

Screenshot:

A screenshot of a phone number

Description automatically generated

A screenshot of a phone number

Description automatically generated

Code for Social Security number regex:

private void jButton2MouseClicked(java.awt.event.MouseEvent evt) {

// TODO add your handling code here:

if (findSocial(txtSocialInput.getText())){

lblTextOutputSocial.setText("Valid Social");

}

else{

lblTextOutputSocial.setText("Not valid Social");

}

}

private Boolean findSocial(String checkThis){

Pattern siocialNumber = Pattern.compile(“\\d{3}\\)?[. -]? \*\\d{2}[. -]?\\d{4}”);

Matcher checkingPossible = sopcialNumber.matcher(checkThis);

return checkingPossible.find();

}

Screenshot:

A screenshot of a phone number

Description automatically generated

A screenshot of a computer

Description automatically generated