Software Development

Project Report



**Lottery Game App**

**Software Development**

|  |  |
| --- | --- |
| **Project Team** | |
| **Student No.** | **Student Names** |
| **x19175108** | **Geneci Ui Fhátharta** |
| **x19139306** | **Rory Cleary** |
| **x19176414** | **Kasia Casserly** |

Contents

[1 Project Overview 3](#_Toc27243676)

[1.1 Project description 3](#_Toc27243677)

[1.2 APP Requirements 3](#_Toc27243678)

[1.2.1 Functional Requirements 3](#_Toc27243679)

[1.2.2 Input-Process-Output (IPO) 3](#_Toc27243680)

[1.2.3 Class Diagram 4](#_Toc27243681)

[2 Section 2: Development 5](#_Toc27243682)

[2.1 Decision on codes: 5](#_Toc27243683)

[2.1.1 LotteryRandom Class 5](#_Toc27243684)

[2.1.2 LotteryGame Class 6](#_Toc27243685)

[2.1.3 LotteryApp Class 7](#_Toc27243686)

[2.2 Testing 9](#_Toc27243687)

[3 Section 5: Project Team 19](#_Toc27243688)

[4 Code and videos used as example: 21](#_Toc27243689)

# Project Overview

## Project description

We were requested to design and develop an application that would allow user to play a lottery game and compare result to the random generated numbers.

## APP Requirements

We understand that user is asked if want to play between 1 to 3 lines. We didn’t manage to meet this requirement. We hard coded 3 lines per game but player can play as many games as player wants.

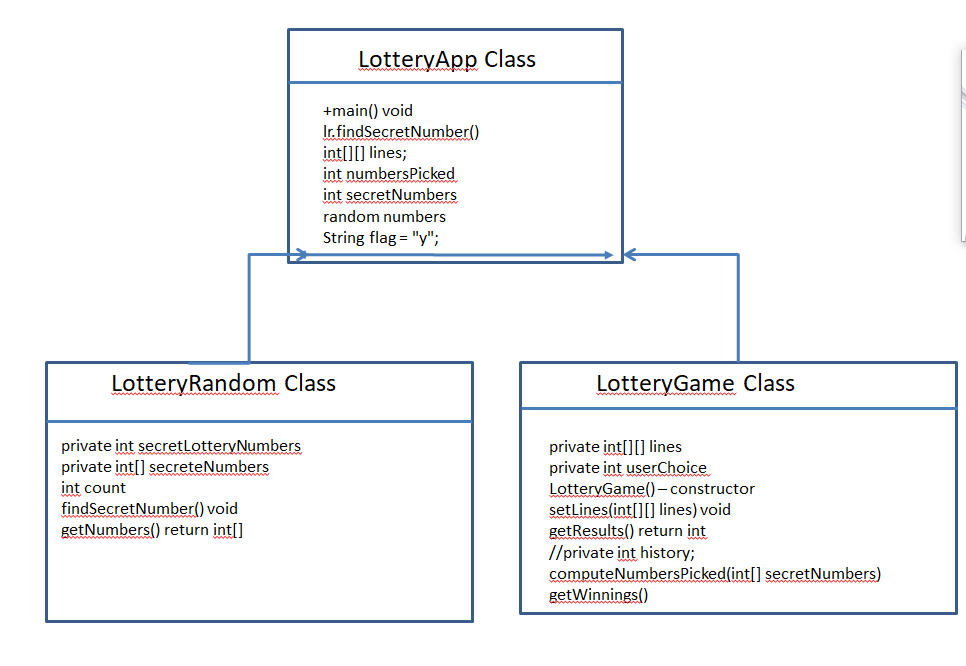
### Functional Requirements

* The app should generate 6 random numbers between 1- 40.
* User should play up to three lines of 6 numbers each.
* User numbers should be compared to the lottery random numbers.
* App should have clear criteria for winners numbers
  + 3 numbers
  + 4 numbers
  + 5 numbers
  + 6 numbers.
* The app should allow user to play the lottery game as many times as they want.
* Display amount of numbers guessed correctly in each line.
* Ask user if want to play again after a game is finished.
* Display history of all games.

### Input-Process-Output (IPO)

|  |  |  |  |
| --- | --- | --- | --- |
| Class / Methods | Input | Process | Output |
| Lottery Random | * Array with 40 numbers | Generate 6 random numbers | 6 random numbers is the secreteNumbers |
| Lottery Game | * Lines with 6 numbers each from user(User Input) * Ask user if want to play again | Generate array with user played numbers  If answer is yes, allow user to play 3 times as one lottery card | Store user numbers into arrays  Numbers played by game card.  User plays 3 lines for each game.  3 lines – 18 numbers generated |
| Lottery App | User numbers  secreteNumbers | Compare arrays to see if any luck for prizes: | Print out result and winner numbers |
| History | Number played by user and games | Generate numbers by user | History contains data for the all line played by user |
| Comparison | User numbers and random numbers | If statements – comparison of the results | Match result to the prizes:   * 3 numbers - 125 * 4 numbers - 300 * 5 numbers – 1,500 * 6 numbers – 150m |

### Class Diagram



# Section 2: Development

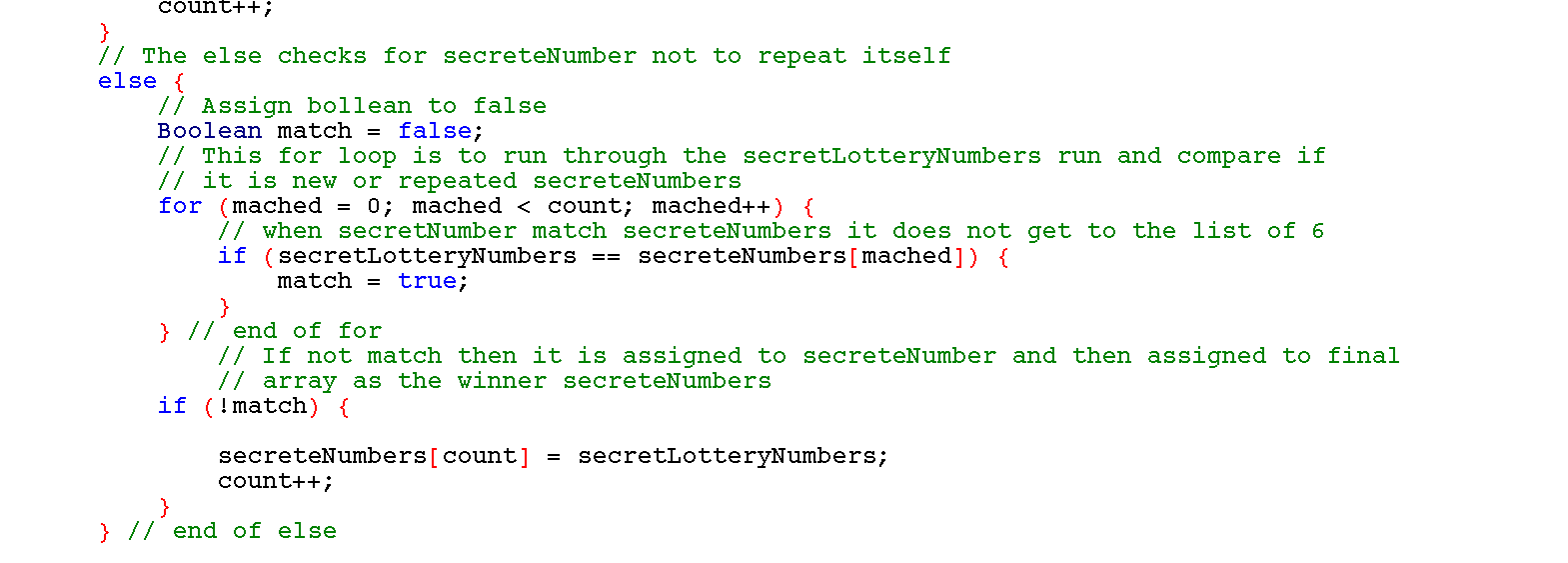
## Decision on codes:

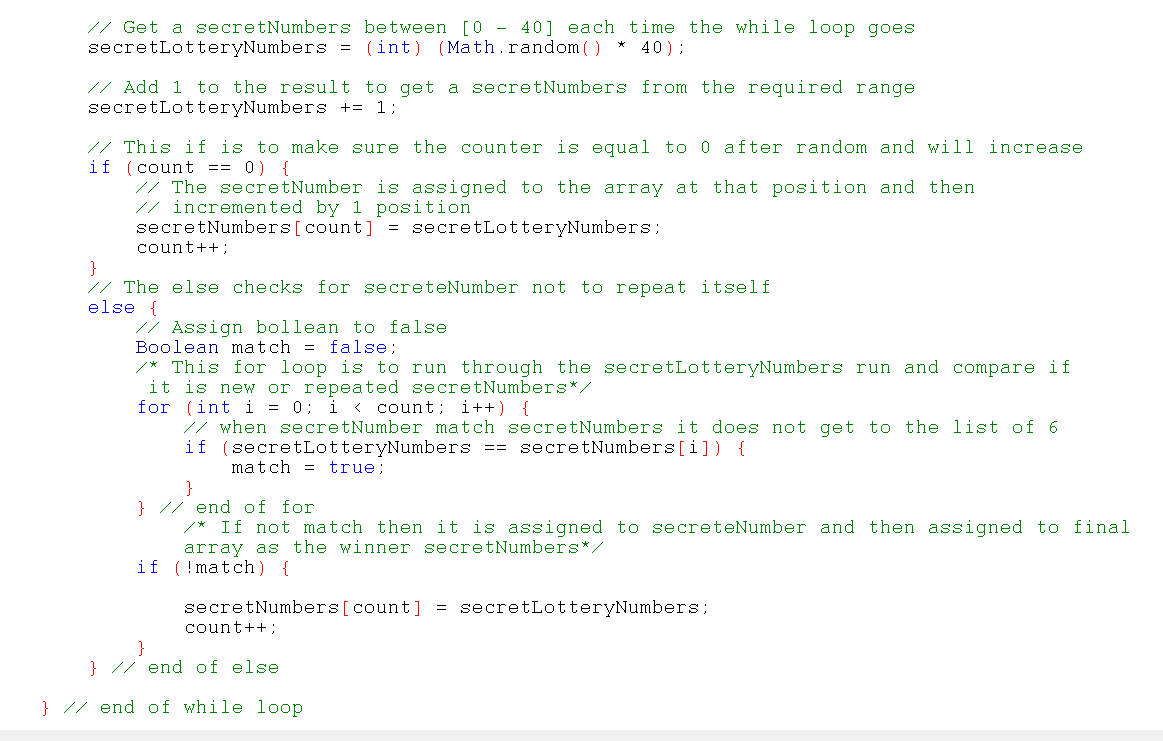
We decided to build at least three classes as below. We want to get a class to generate the random numbers. Another class to hold numbers entered by users, and the app class to compare and generate the history of the game.

### LotteryRandom Class

#### Coding

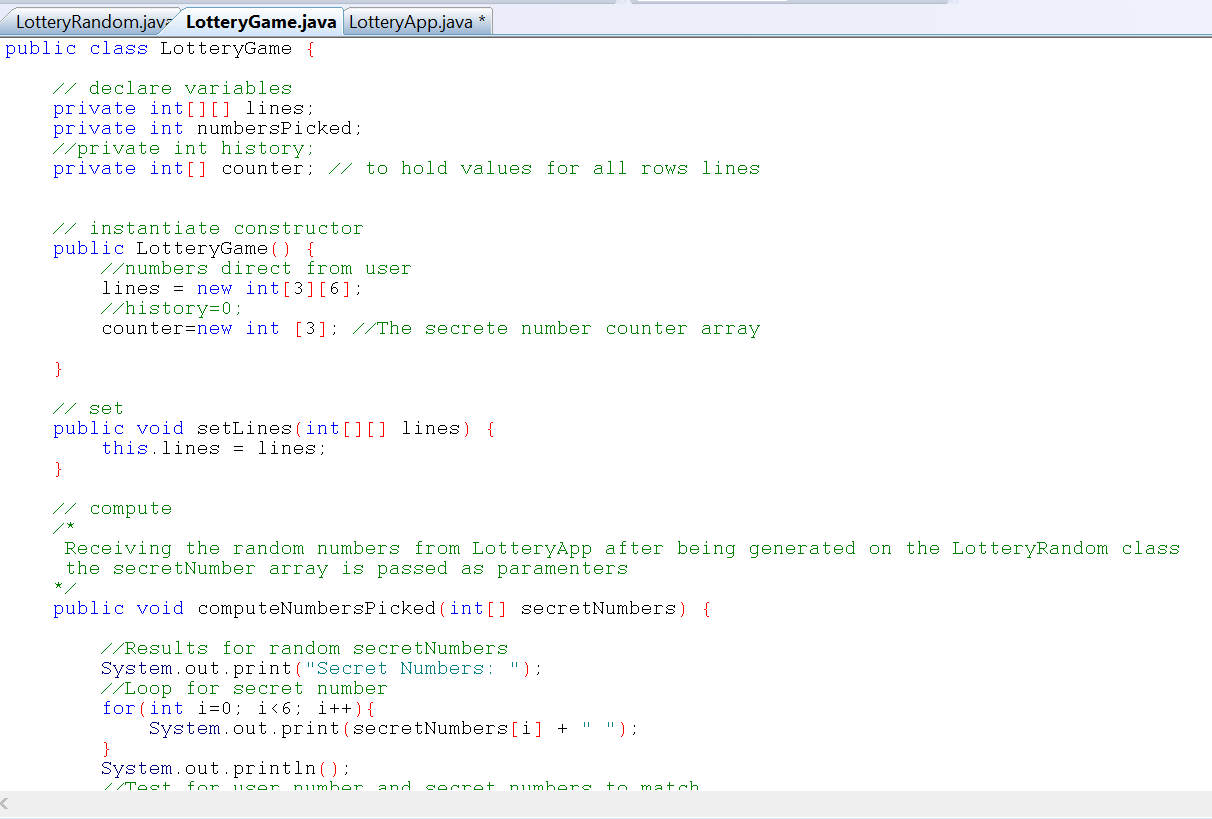
This class was first designed only to generate the random numbers. Then we realized that those numbers were repeating it sometimes. We developed code to eliminate the repetition.

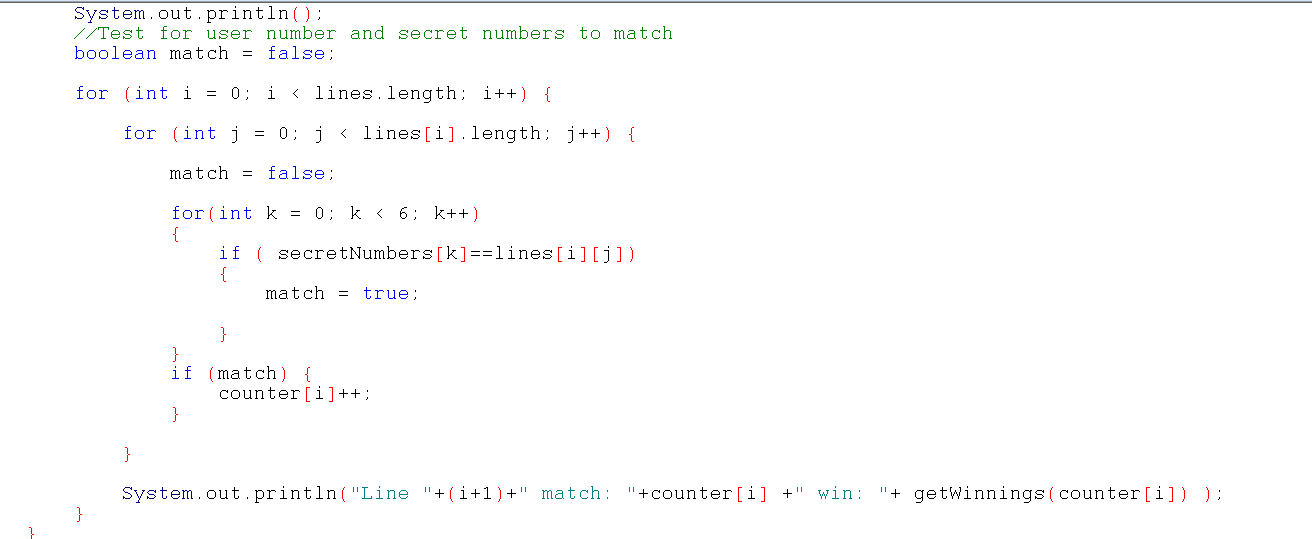


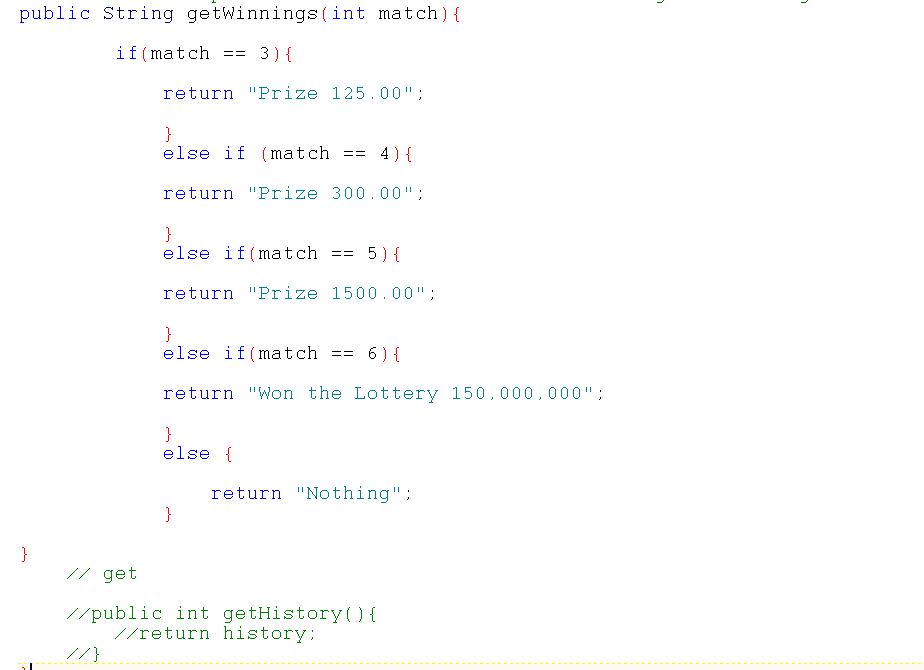


### LotteryGame Class

This class was designed to take user numbers and publish them in the app class. It compares picked numbers to secret numbers using if statements and match with the prizes.



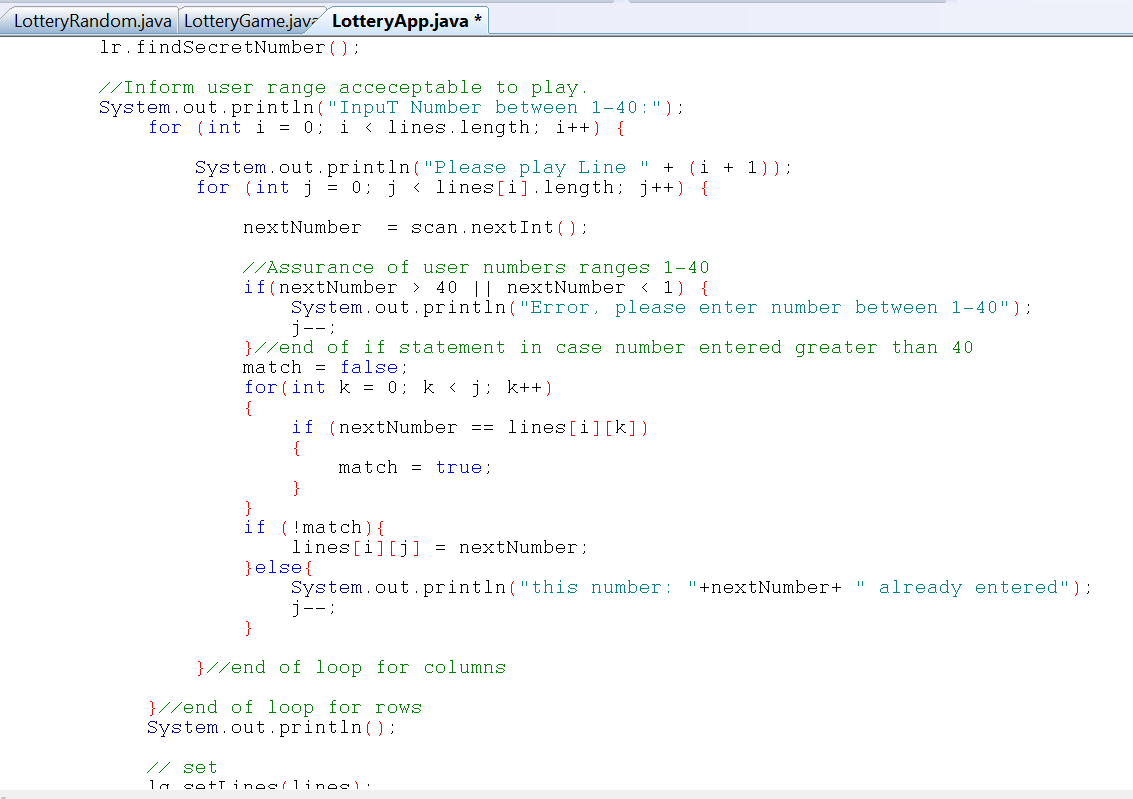


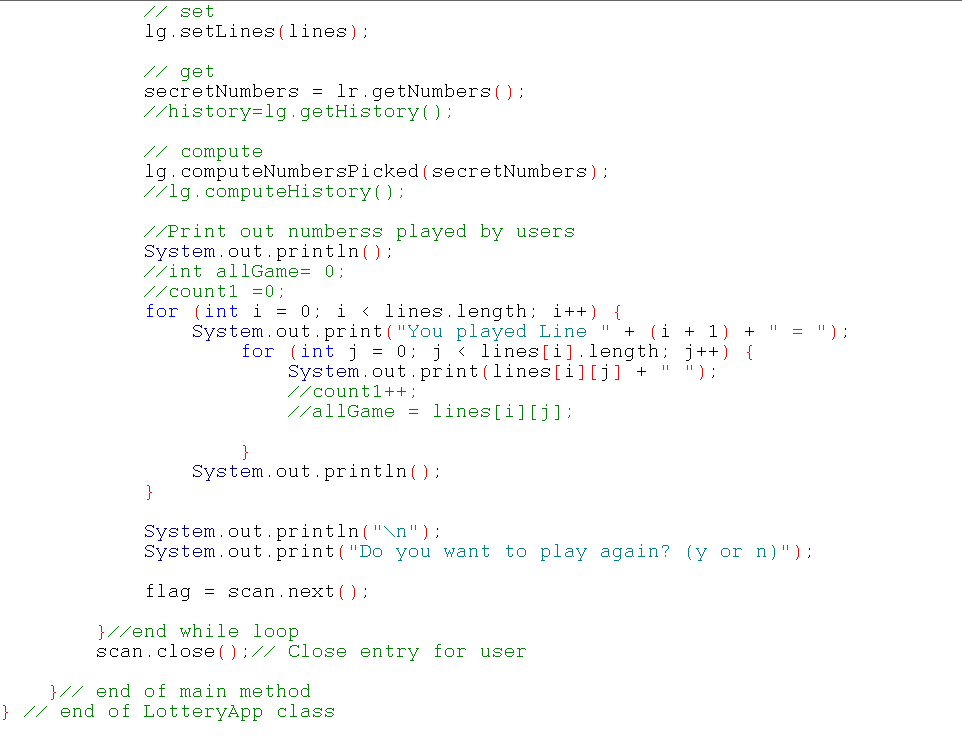


#### Coding

### LotteryApp Class

LotteryApp obtain all the inputs and outputs which have been processed from LotteryRandom Class and LotteryGame Class.

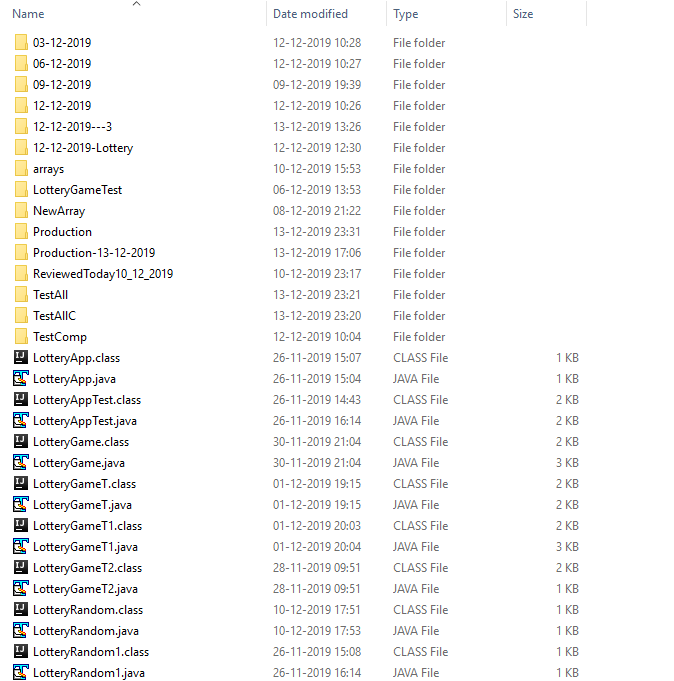




#### Coding

## Testing

We decided to test as we developed. Below you see an example of the folders quantity.



Nothing printed out

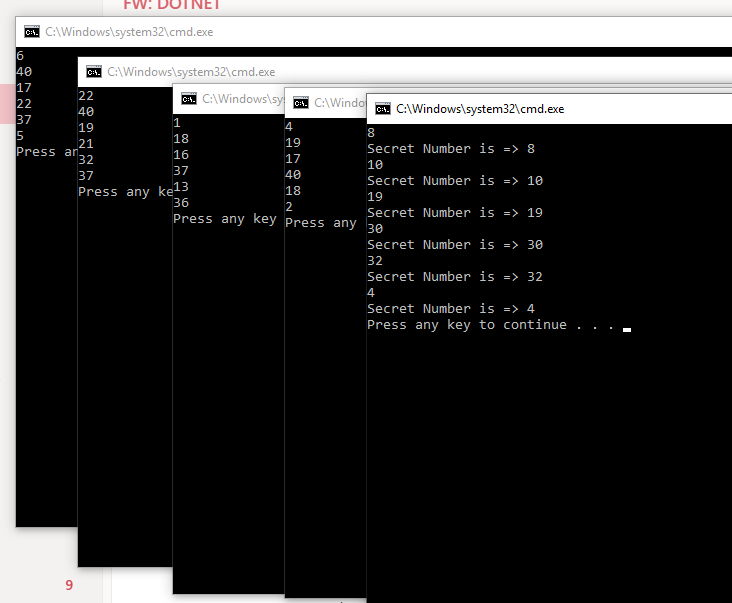
Machine generated alternative text: C:’’.indcc :.tE--L:_ Cr:itE:E — Q X
random between
= randomNumbers.
to eve rytime the
/1 assign value from
secretNumbers[ count]
}
numbers to array
= numbers;
I/Output results
System.out.println(”The secret numbers are: “ + numbers);
}//end main method
}//end class App
I/Create and define
secretNumbers = new
Random randomNumbers
I/Create a while loop
while(count <
I/Create
numbers
I/Add 1
numbers += 1;
obj ects
int[6];
= new Random();
to go through 6 times
1—40
nextlnt(40);
loop passes
from 1 to 40

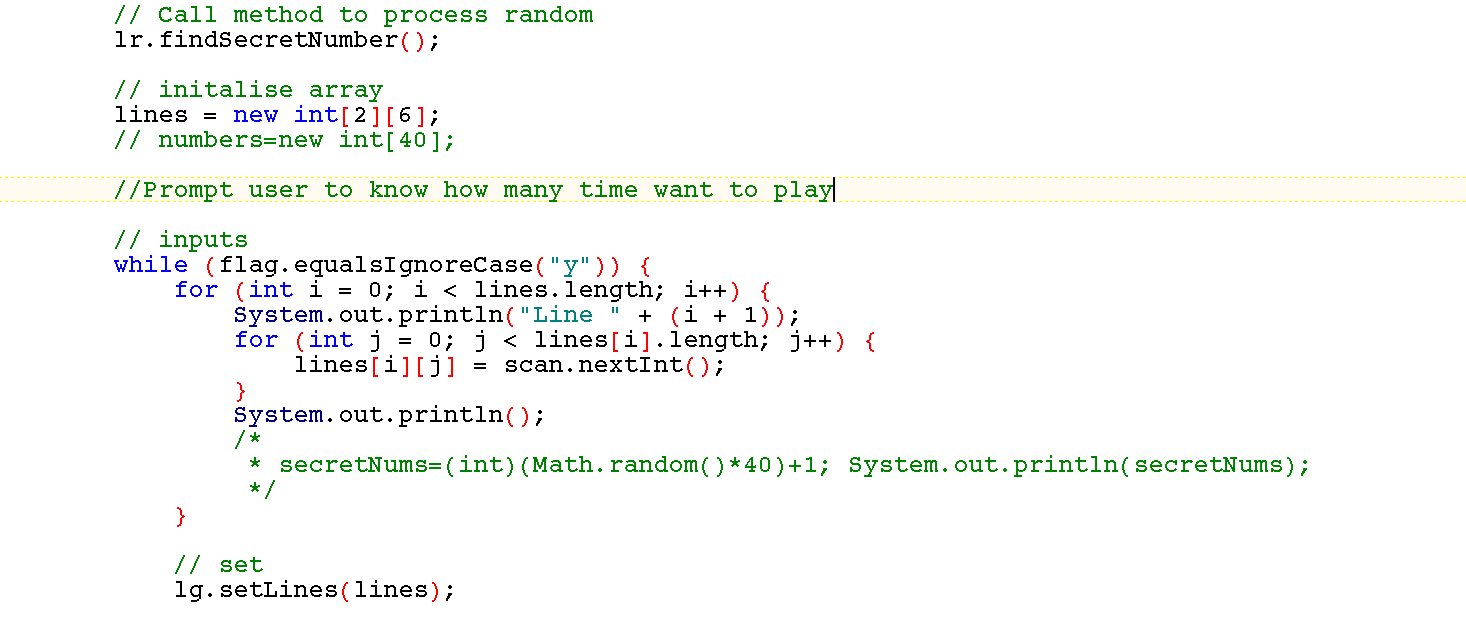
Added system print out inside the loop and got an Infinite Loop

Machine generated alternative text: Lotteryapp.java
he secret numbers are: 28
2 1 secret numbers are: 1
- secret numbers are: 7
22 he secret numbers are: 11
23 I/Create and define objects secret numbers are: 30
- secret numbers are: 5
24 secretNumbers = new int[6]; he secret numbers are: 16
25 Random randomNumbers = new Random() he secret numbers are: 20
he secret numbers are: 13
26 I/Create a while loop to go through 6 times he secret numbers are: 37
he secret numbers are: 12
27 while(count < 6){ he secret numbers are: 38
28 I/Create random between 1 — 40 he secret numbers are: 31
In secret numbers are: 29
29 numbers = randomNumbers. nextlnt( 40); he secret numbers are: 6
30 I/Add 1 to everytime the loop passes from 1 to 40 he secret numbers are: 27
he secret numbers are: 6
31 numbers += 
L, he secret numbers are: 40
32 he secret numbers are: 27
he secret numbers are: 39
33 II assign value from numbers to array he secret numbers are: 24
34 secretNumbers[count] = numbers; he secret numbers are: 25
he secret numbers are: 11
35 he secret numbers are: 15
. j secret numbers are: 13
36 System. out.println( “The secret numbers are: “ + numbers), he secret numbers are: 16
37 } he secret numbers are: 19
he secret numbers are: 20
38 he secret numbers are: __________ vi
39
40 I/Output results
41 System.out.println(”The numbers are: “ + numbers + “secret numbers” + secretNumbers);
42
43 }//end main method
44 }//end class App

Created a Boolean to check for repetitive numbers

Got an array out of bound

Machine generated alternative text: .L
Boolean match = false; .
C:\Windows\system32\cmd.exe — D X
The n,tj  aje: 9 A
for(int j = O; j < count; j’he secret numbers are: 24
7he secret numbers are: 19
Exception in thread “main” java.lang.ArraylndexOutOfBoundsException: 6
if (numbers == secretNumbers[ j]) { at LotteryApp.main(LotteryApp.java:52)
‘ress any key to continue . . .
match = true;
).
if (!match){
secretNumbers[count] = numbers;
count ++;
).
}
}
I/Output results
System.out.println(”The secret numbers are: “ + numbers);
).
}//end main method
}//end class AppMachine generated alternative text: I/Used while loop to get run six time and no increment is needed
while( i < num.length){
/1 Get a number between [0 - 40] each time the while loop goes
secretNumbers = randomNumbers.nextlnt(40);
/1 Add 1 to the result to get a number from the required range interacting with loop
secretNumbers += 1;
L I/This if is to make sure the counter is equal to 0 3
L- (i==0){
C rLi* crcLE — D X
num[i] Secret Number is 1 => 11 A
Secret Number is 2 => 15
ret Number is 3 => 24
‘h, ret Number is 3 => 11
ret Number is 4 => 28
ret Number is 5 => 32
else’ ret Number is 6 => 17
t. s any key to continue .
Boolean
for(int
ifMachine generated alternative text: I/The else chec]cs for secreteNumber not to repeat itself
else{
I/Assign bollean to false
Boolean match = false;
I/This for loop is to run through the secretNumbers run and compare if it is new or repeated number
for(int j = O; j < i; j++){
I/when secretNumber match num it does not get to the list of 6
if (secretNumbers == num[j ] ){
match = true;
).
).
I/If not match then it is assigned to secreteNumber and then assigi
if (!match){
num[i] = secretNumbers;
}
System.out.println(”Secret Number is “ ÷ i + “ => “ + secretNumbers);
}
}
}
umbersMachine generated alternative text: //int
mached;
C:\Windows\system32\cm...
?cret Number is => 23
?cret Number is => 13
?cret Number is => 22
?cret Number is => 8
?cret Number is => 38
?cret Number is => 25
he secret numbers : 23
he secret numbers : 13
he secret numbers : 22
he secret numbers : 8
he secret numbers : 38
he secret numbers : 25
ress any key to continue .
D I  C:\WindowAsystem32\cmd.... — D X
Sectt Numbt, => 29 A
Secret Number is => 33
Secret Number is => 4
Secret Number is => 32
Secret Number is => 17
Secret Number is => 1
The secret number : 1==> 29
The secret number : 2==> 33
The secret number : 3==> 4
The secret number : 4==> 32
The secret number : 5==> 17
The secret number : 6==> 1
. . Press any key to continue . .
4
5
E
7
8
2
3
I/Create and define objects
numbers = new int[6];
LotteryRandom ir = new LotteryRandom();
//Caii method to process caicuiation
ir. findSecretNumber();
I/Get vaiues returned
secretNumbers = ir. getSecretNumber();
numbers = ir.getNumbersQ;
for(int i=O; i<6; i+÷){
I/Output resuits
System. out.printin( “The secret number :
}
}//end main method
}//end ciass App
5
6
7
8
9
‘o
1
A
“ + (i-i-1) + “ ==> “+numbers[i]);
V



Test for range numbers

Machine generated alternative text: for (j = 0; j < lines[i].length; j++) { C:r: :::tErhcnz:EE — D X
lines[i] [j]
= scan.nextlntQ;
1/Assurance
if(lines[i][j]
}//end
ranges 1-40
lines[i][j] <
in case number
1) {
entered greater than
40
}//end
}//end of
System. out. println();
/1 set
1g. setLines( lines);
11 compute
1g. computeNumbersPic]ced();
¡11g. computeHistory();
of user numbers
>4° II
System. out.println( “Error,
of if statement
of
loop
please enter number between 1-40”);
loop
for
for columns
rows

The lines, history and secret numbers are printing all together.

When one member of the team was testing a code designed by the other member encountered an error.

Commented count and run test. The system compiled but then we got an error

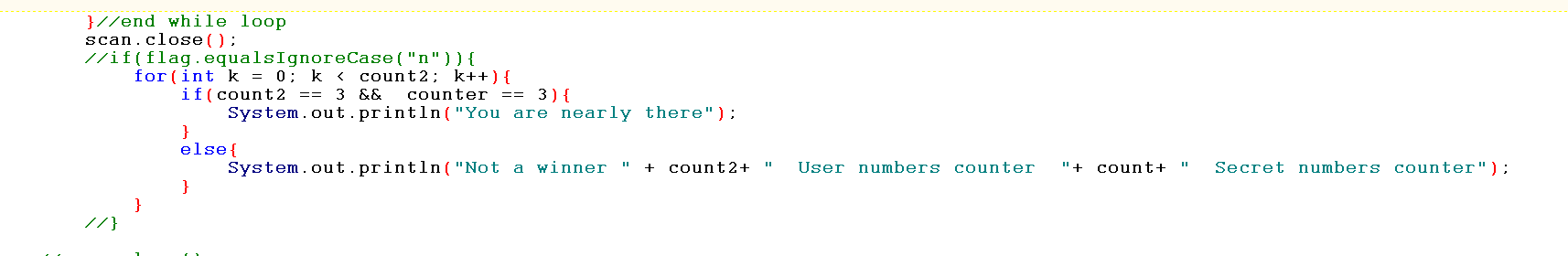


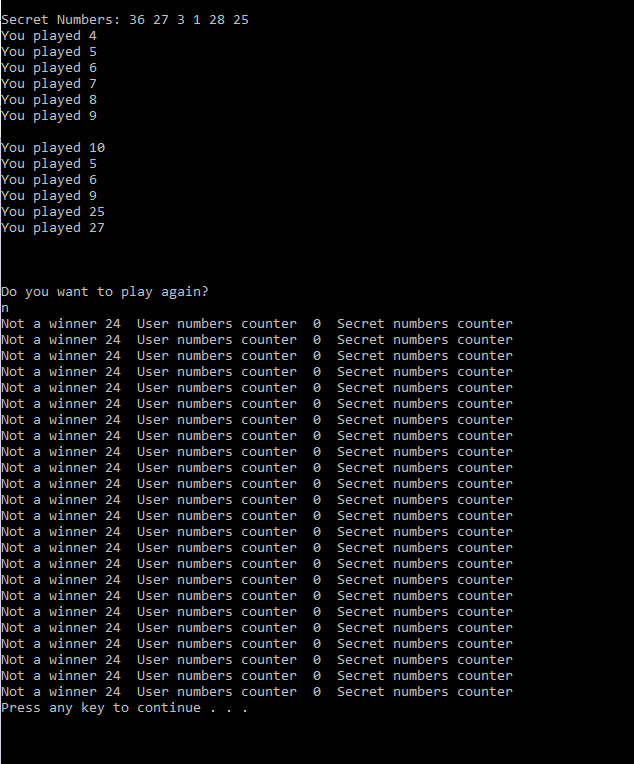
Commented line of code that is not necessary and it compiled and worked ok

Machine generated alternative text: LotteiyGame.java LotteryRa n dom a a IotteryApp.java
52 System.out.println(”Error, please enter number between 1-40”);
53 1——;
54 }Ilend of if statement in case number entered greater than 40
55 }Ilend of loop for columns
56 }Ilend of loop for rows
57 System. out. println();
58
59 Ilfor(int i=0;i<lines. length; i++){
60 Ilfor(int j=0; j<lines[i] . length;
61 //if(lines[i]==lines[j]){
62 //counter++;
Illines[i][j]=scan.nextlntQ;
64 /1)
65 1/)
66
67
/1 set
69 1g. setLines( lines);
70
71 11 compute
72 1g. computeNumbersPic]ced();
73
74 11 get
75 numbersPicked = 1g. getNumbersPickedQ;
76 secreteNumbers = lr.getNumbersQ;
77 /11g. computeHìstory();
78
79

Troubleshooting comparison for code suggested in the Computer Support. We got the counter printing as total but not what we wanted yet.

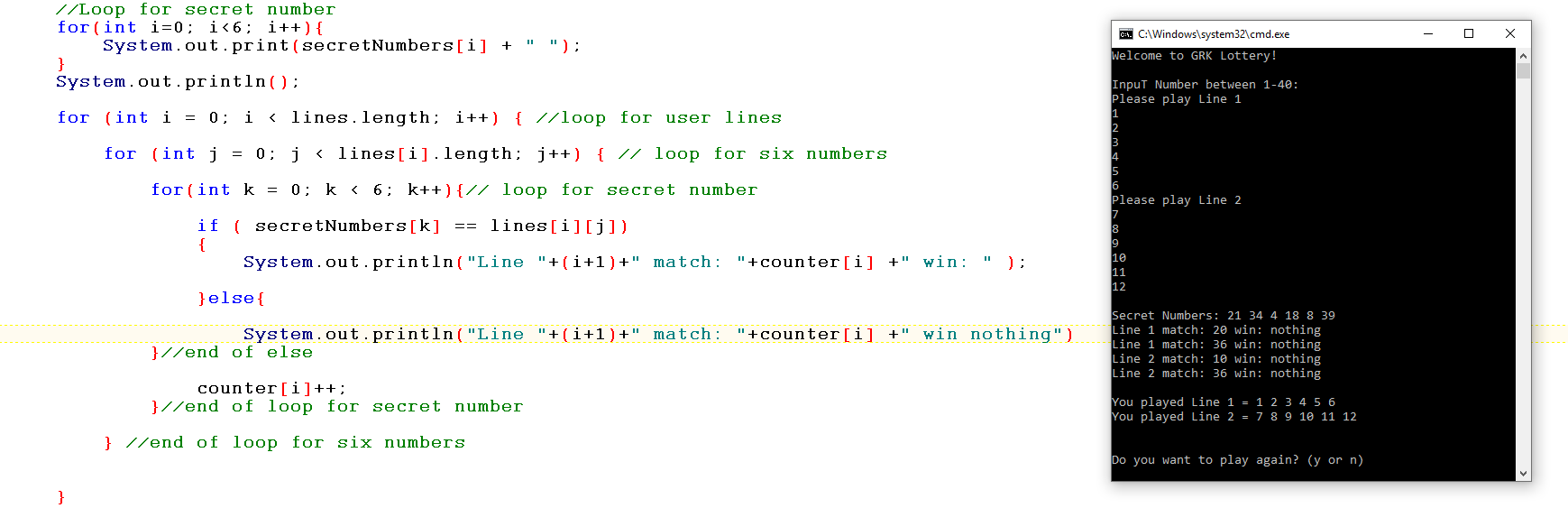
At this pointing we were trying to do everything in the App class.



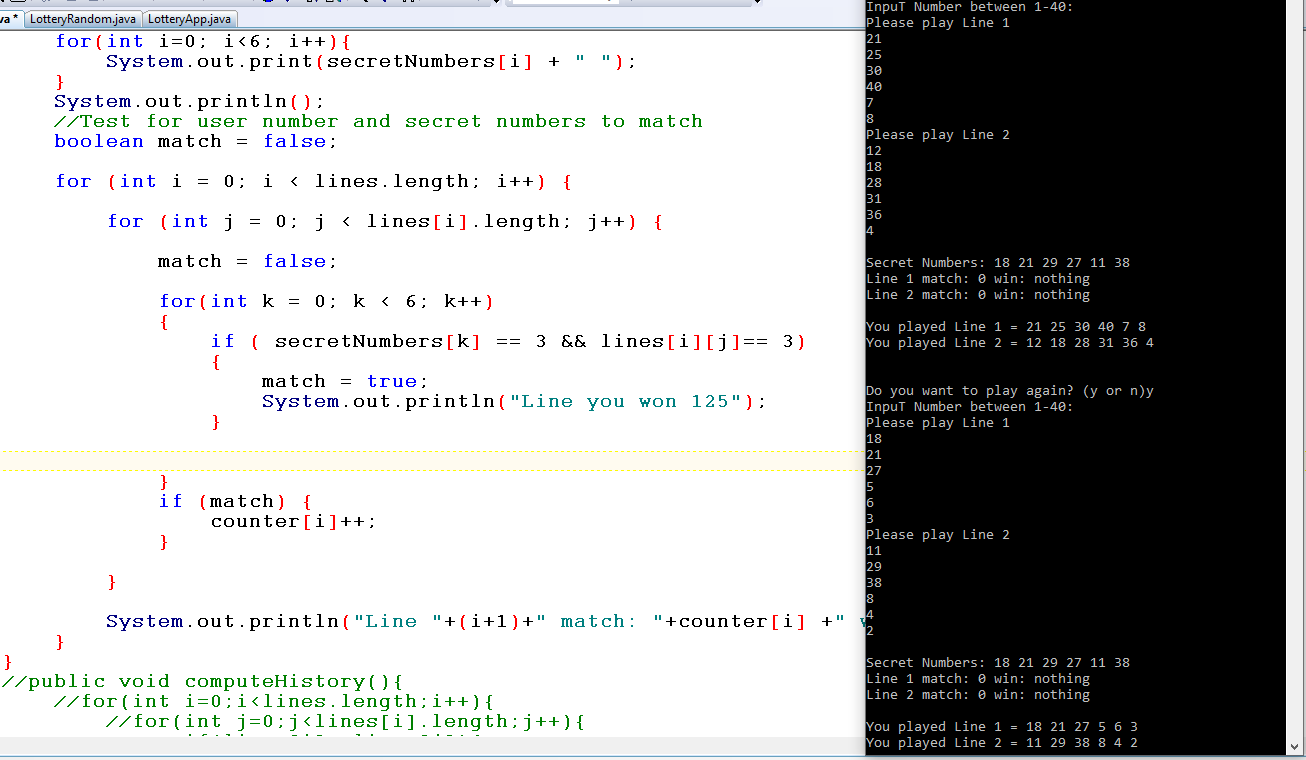


When moving the secretNumber as parameters to the LotteryGame class, we found the missing piece.

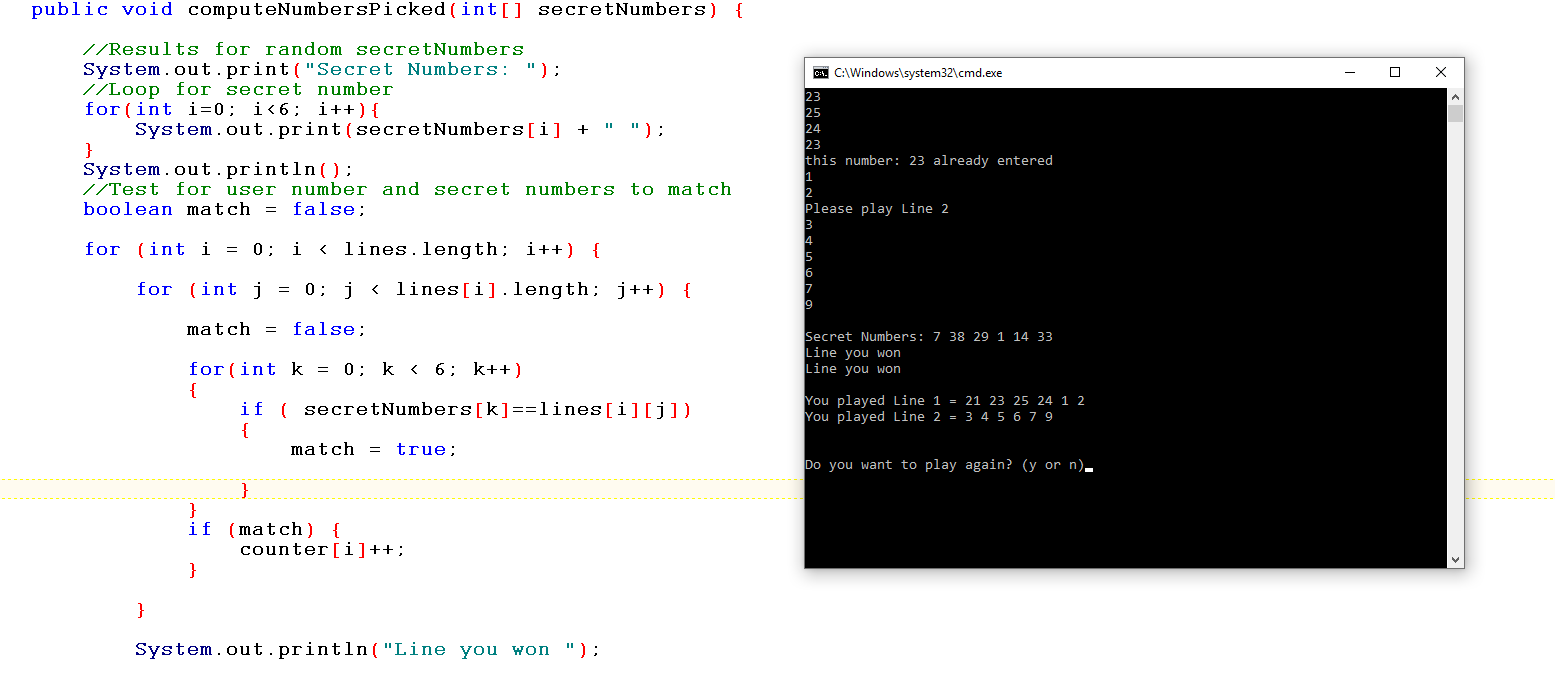
Test 14/12/2019 after passing the secret numbers as parameters to the LotteryGame class



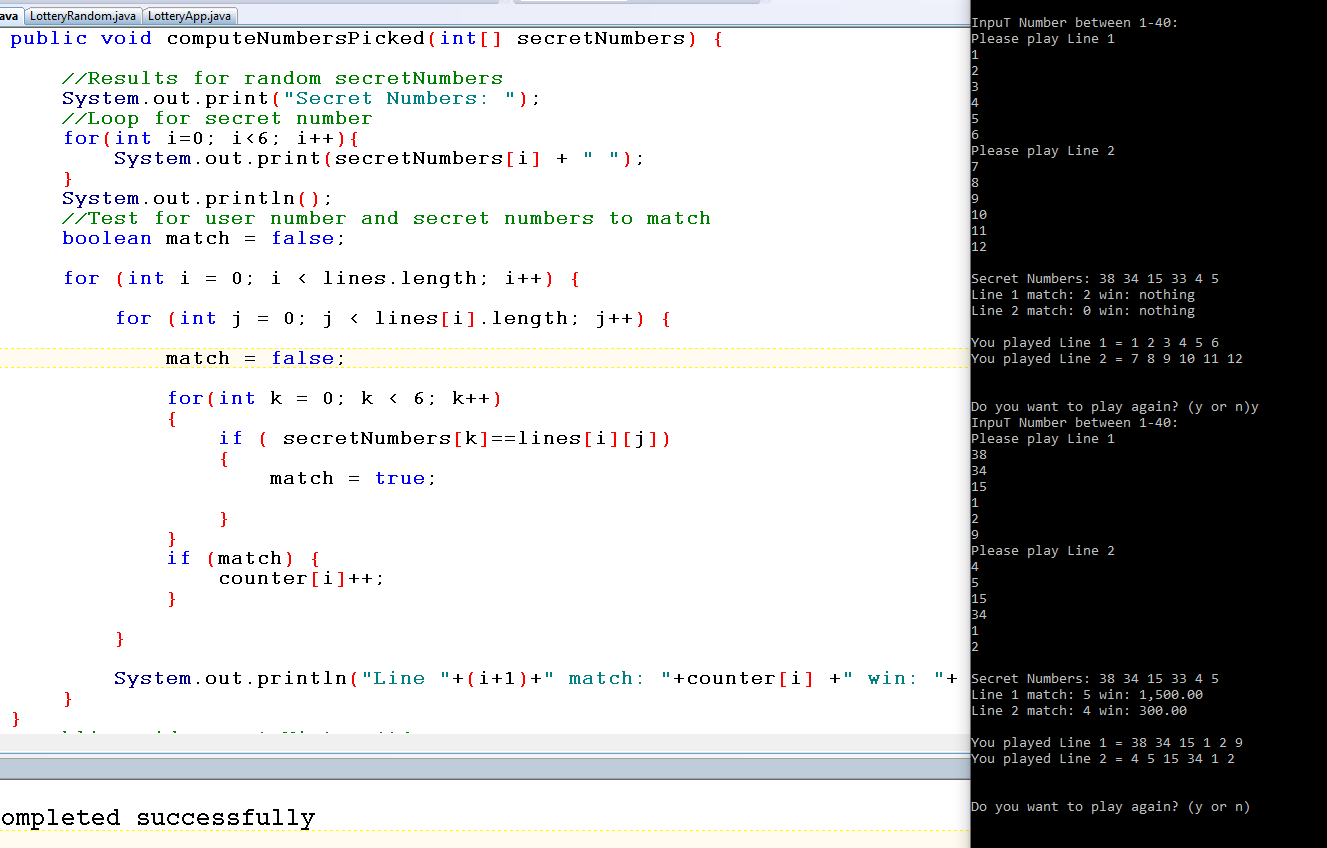
We created a Boolean to check matched numbers. We got it working but the result was not accurate yet, even when we set second game to have three lottery numbers.



We then tested only for match true and the sentence worked fine, but the result was not correct.

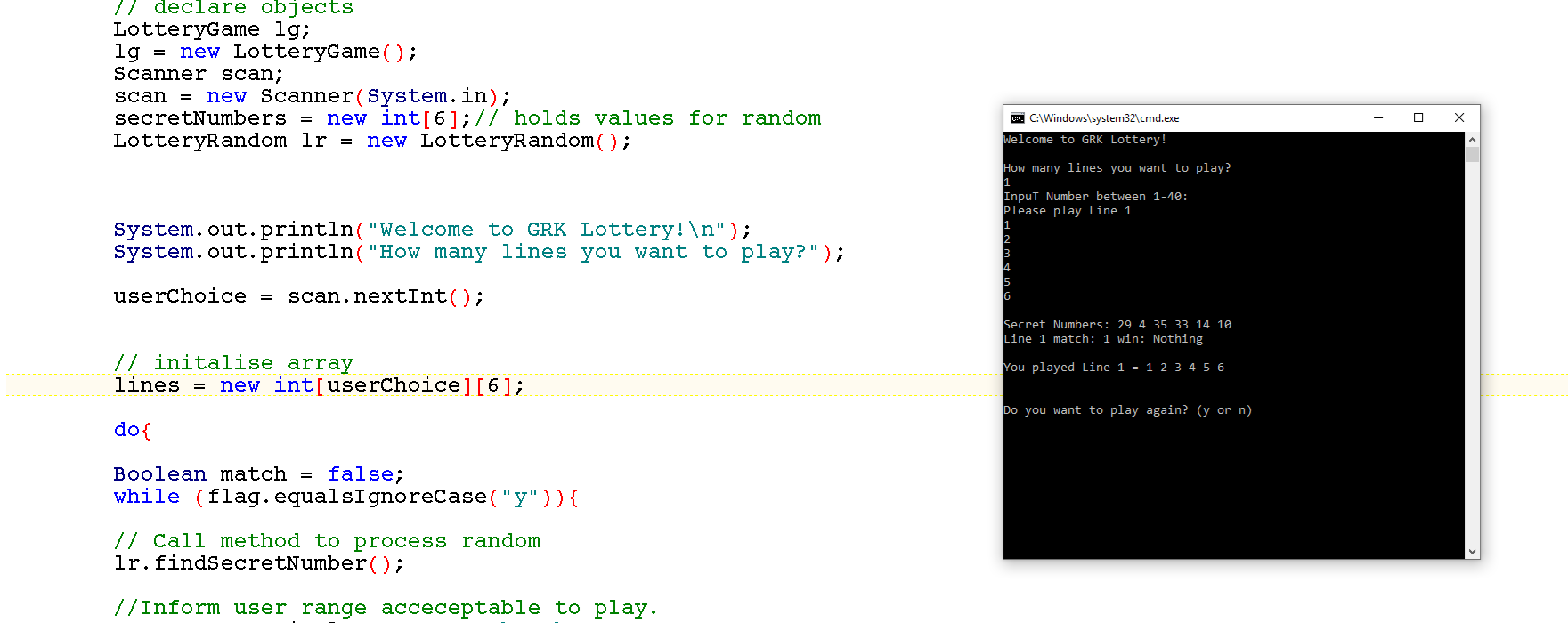


Created a method to hold table result and called the method in the print line.

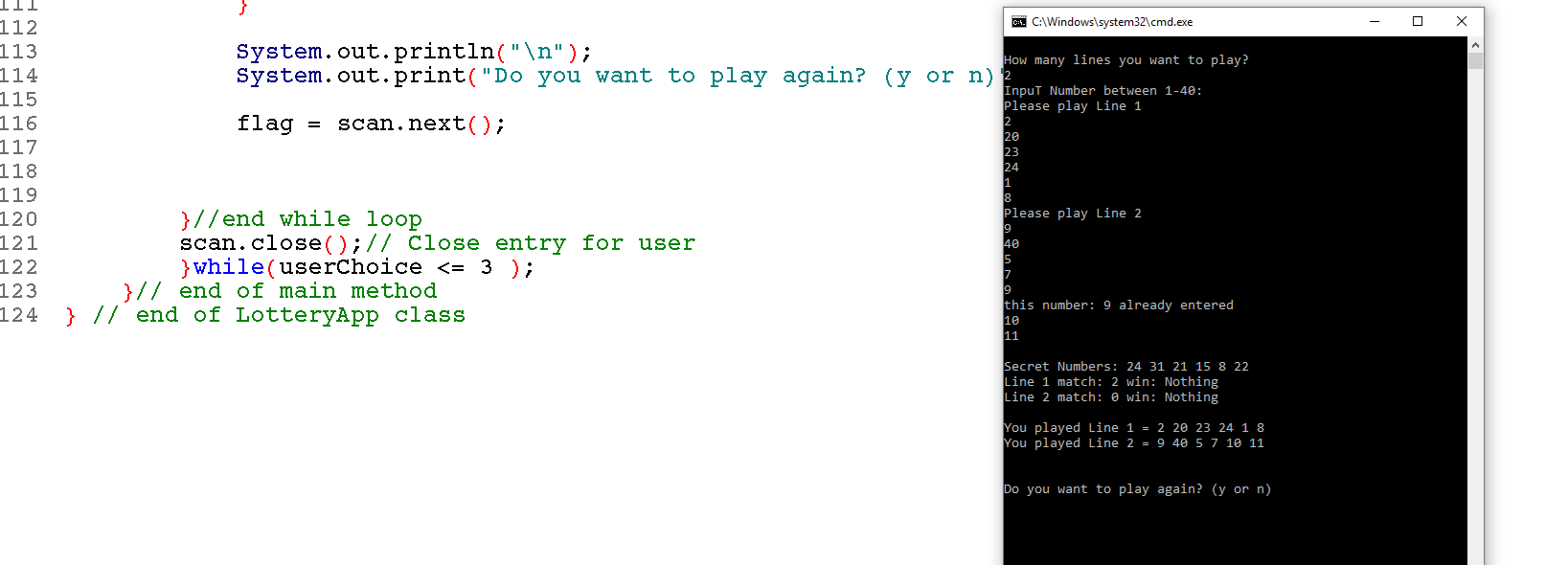


Adjusted LotteryGame to ask how many lines the user wants to play tested for:

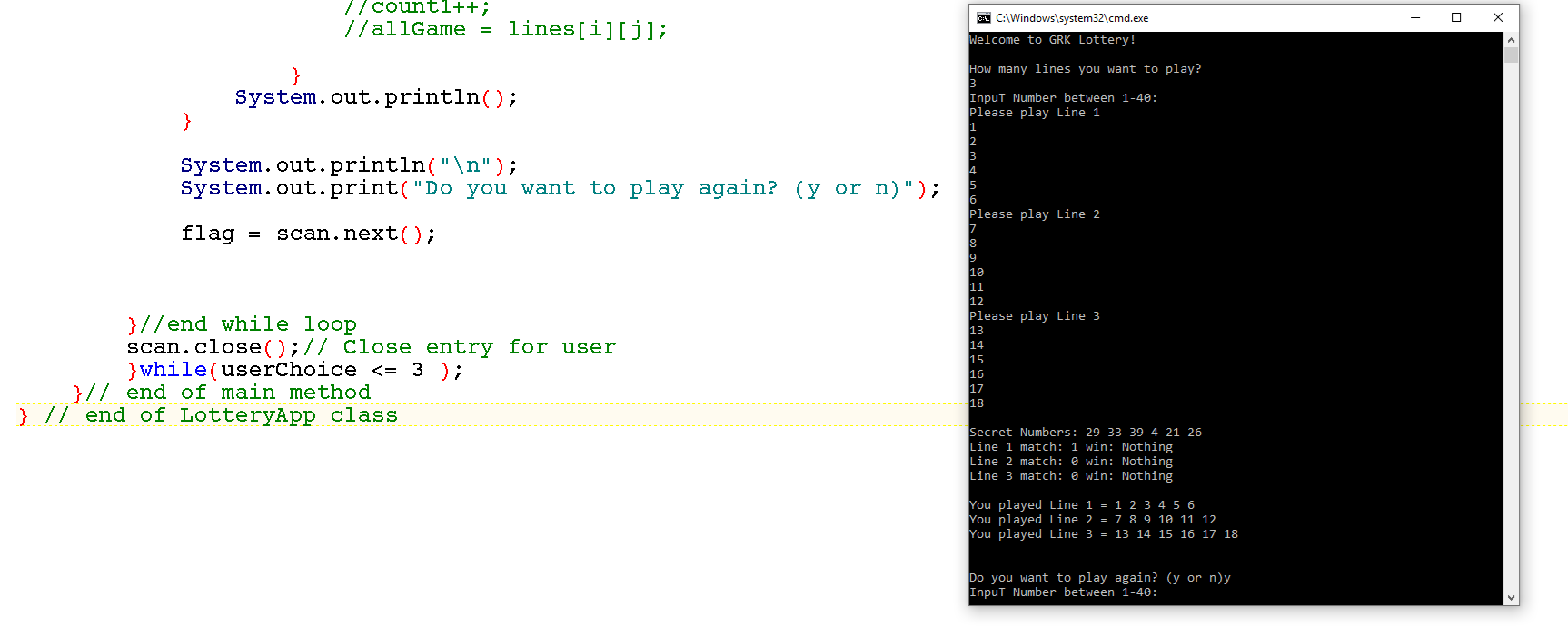
1 line – Passed



2 lines – Passed



3 lines - Passed



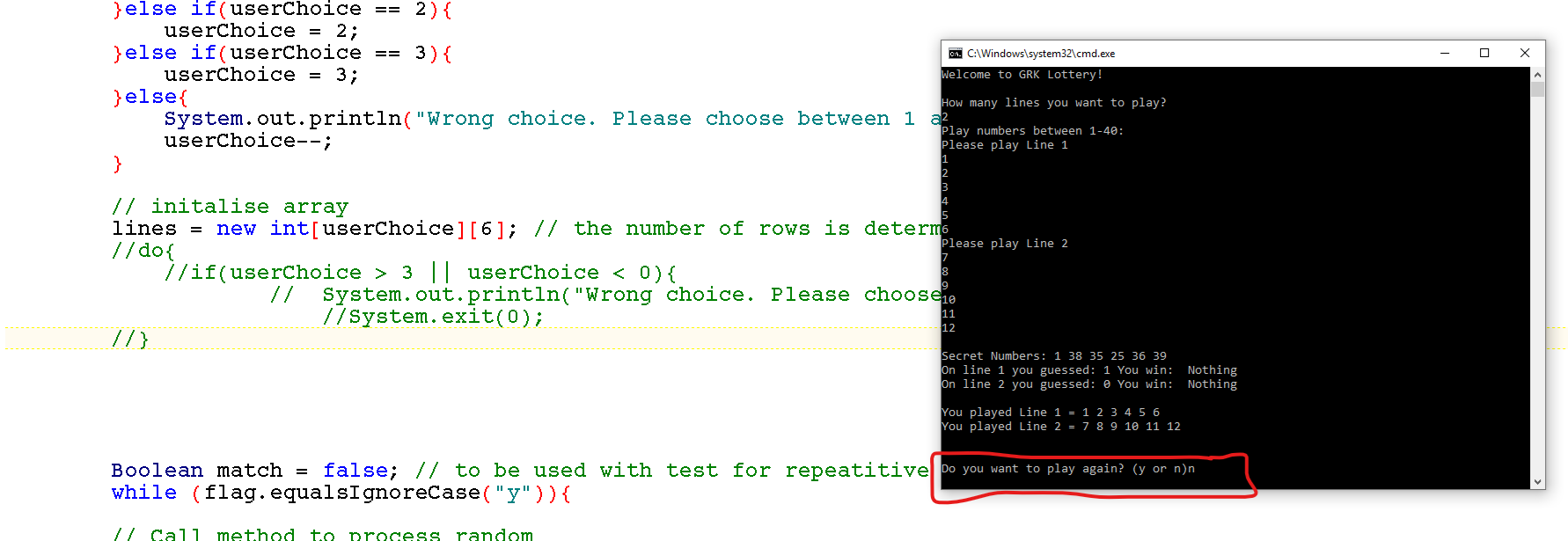
Up to this point we had not tested for lines above 3. When we did it failed. So we restarted process of testing.

The app failed for over 4 in many aspects and we kept changing code until found the right solution.

With the do/while loop if we choose n the app didn’t close. We used many option such as:

* break;
* userChoice --;
* System.exit(0);

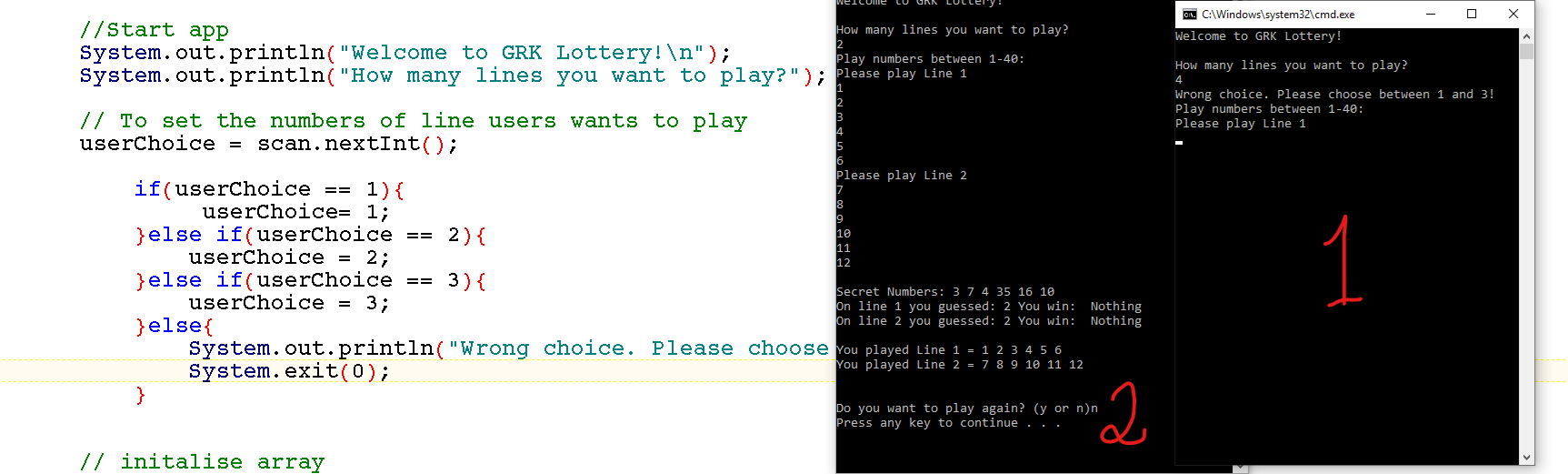
Nothing worked.



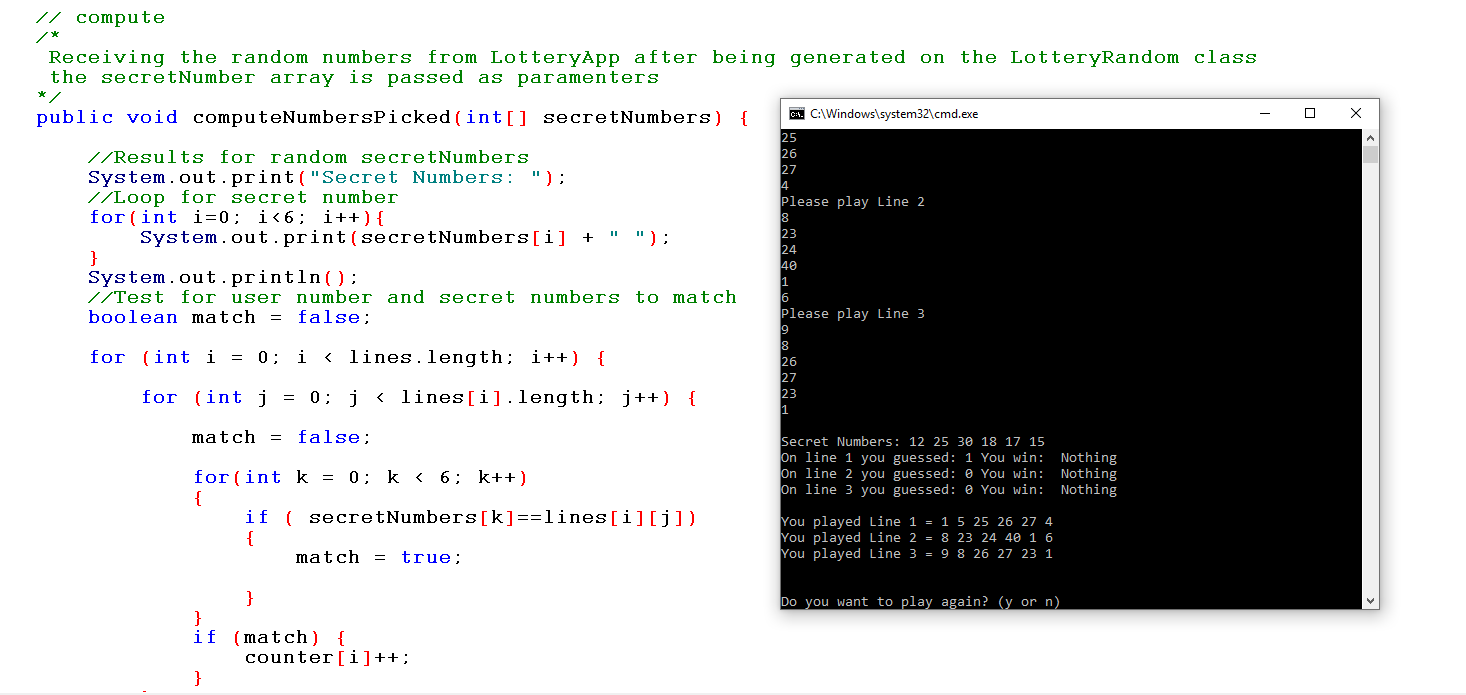
We decided to use a simple if statement assigns the user choice to itself and them the solution was complete.

Number 1 – Failed with while loop as it was not giving opportunity to restart game

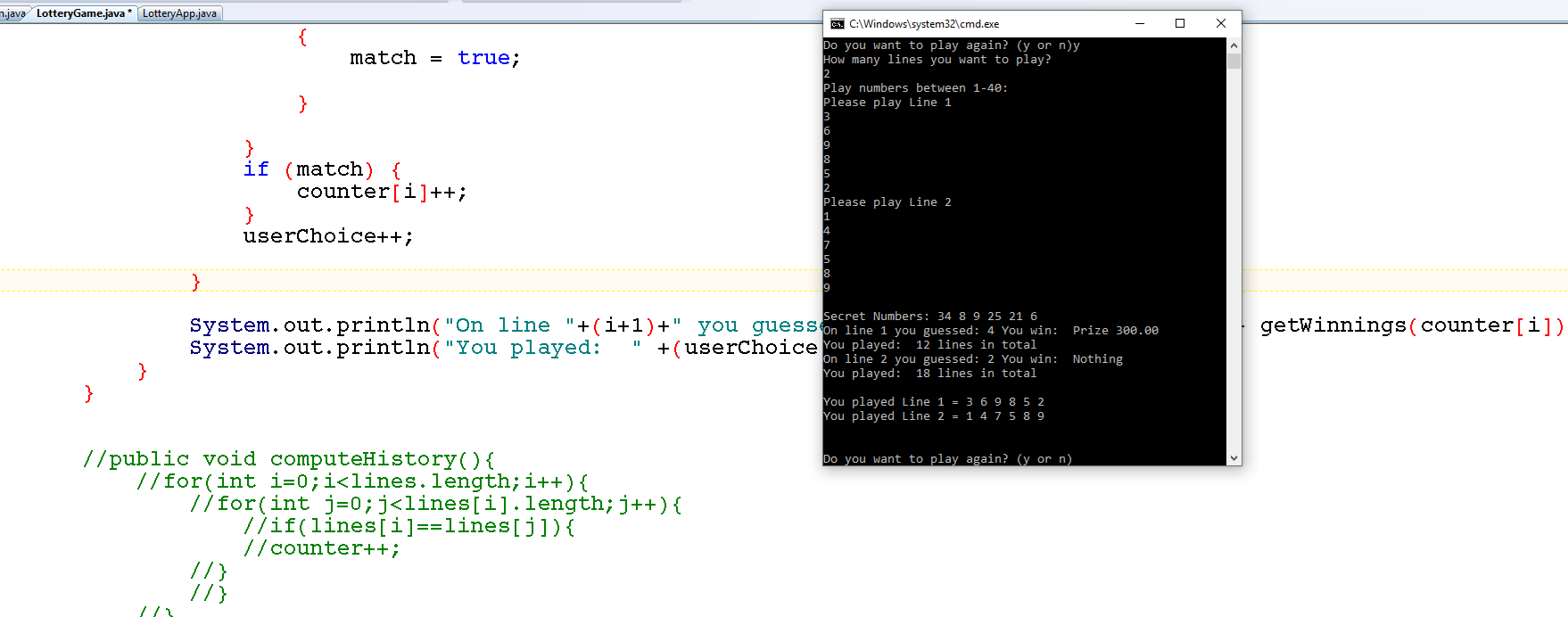
Number 2 – The if statement passed. All good.



User receipt – Passed for number of lines played and number guessed



We worked until last minute to get history working. At the end we got the total lines played.



Divided total number by 6 and got total lines played

Machine generated alternative text: {
match = true
,
).
C: ‘.Hndc. Ht EE C X
A
3
if (match) { lease play Line 3
counter[i]++; s
}
userChoice++’
,
59
Error, please enter number between 1-40
}
System. out.println( l’on line “÷(i÷l)÷” you guessed: “÷counter[i] ÷1’ y ?cret Numbers: 310 274064
line 1 you guessed: 3 You win: Prize 125.00
line 2 you guessed: 1 You win: Nothing
} line 3 you guessed: 1 You win: Nothing
ou played: 5 lines in total
. ouplayedLinel=369852
System.out.println(”You played: “ ÷(userChoice /6)+ “ lines in total ), ouplayedLine2=147202123
) ou played Line 3 = 25 26 24 269 29 40
ou played: 2 lines in total
you want to play again? (y or n)_
I/public void computeHistoryQ{
//for(int i=O;i<lines.length;i÷÷){ ___________________________ j
//for(int j=O;j<lines[i].length;j+÷){
//if(lines[i]==lines[j ]){
/1 counter++;

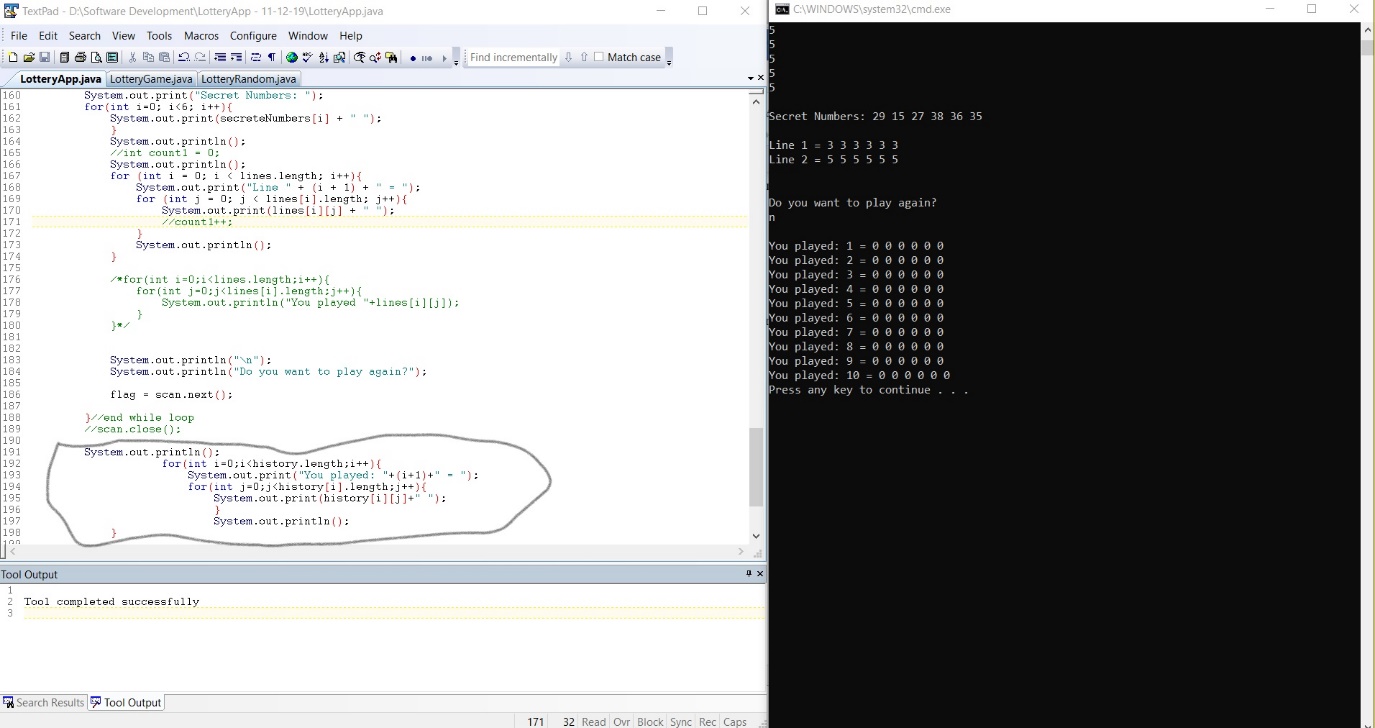
# Section 5: Project Team

We decided that each one of us would code one class and that everyone would help to improve each other class as needed.

As this is a very tricky project and one class got more complex than others. We decided to have folders in the GitHub with classes’ version. In this case anyone fixed some issue would upload a new version of the project and we all would contribute to it.

In the original stage each student created their version of the game and random classes. During the second stage we combined best elements from the each class and used code which were working correctly.

In the later stages we worked together to create final project requirements like comparison, elimination of the duplications and historical data.



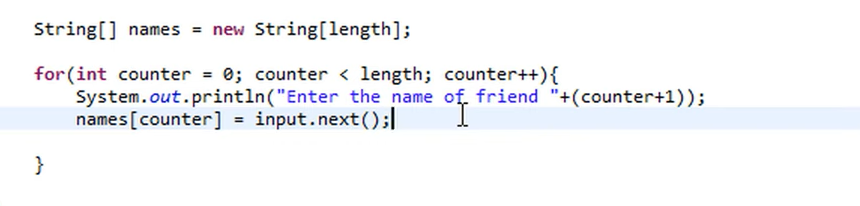
Due to time constrains and complexity of the project we ran out of time to fully implement history aspects of the project.

# Code and videos used as example:

<https://codereview.stackexchange.com/questions/202521/lottery-in-the-command-line>

<https://www.youtube.com/watch?v=Q56GaO5FJdk>

<https://www.youtube.com/watch?v=RTvTcpvhcl4>



<https://www.youtube.com/watch?v=SE5Oc89JumQ>

Used to create methods with parameters in the LotteryGame

