CPSC 1150 – Assignment 3

By Cordell Bonnieux

Count

Filename: Count.java

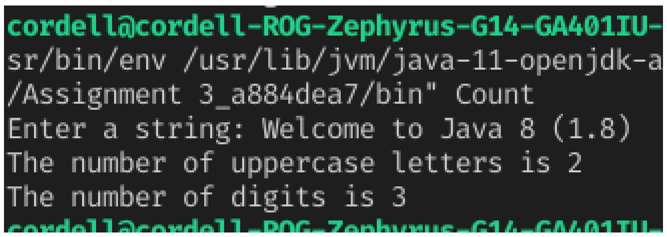
This program reads a string from the user, then iterates over each character determining if it is a upper-case letter or digit. If so, the appropriate counter is incremented, if not the loop continues to the next item.

Once the string has been processed, the program prints the number of uppercase letters and the number of digits to the console.

Pseudocode:

```
START
PRINT “Enter a string”
READ userString
COMPUTE integers digitCounter, uppercaseCounter
FOR integer i = 0 AS LONG AS i <= userString.length THEN i++
    IF userString at index i (ASCII value) >= 65 AND  userString at index i (ASCII value) < 91
        COMPUTE uppercaseCounter++
    ELSE IF userString at index i (ASCII value) < 58 AND  userString at index i (ASCII value) >= 48
        COMPUTE digitCounter++
PRINT “The number of uppercase letter is “ uppercaseCounter
PRINT “The number of digits is “ digitCounter
END
```

Test Case:



```
cordell@cordell-ROG-Zephyrus-G14-GA401IU-
sr/bin/env /usr/lib/jvm/java-11-openjdk-a
/Assignment 3_a884dea7/bin" Count
Enter a string: Welcome to Java 8 (1.8)
The number of uppercase letters is 2
The number of digits is 3
```

SSN – Social Security Number

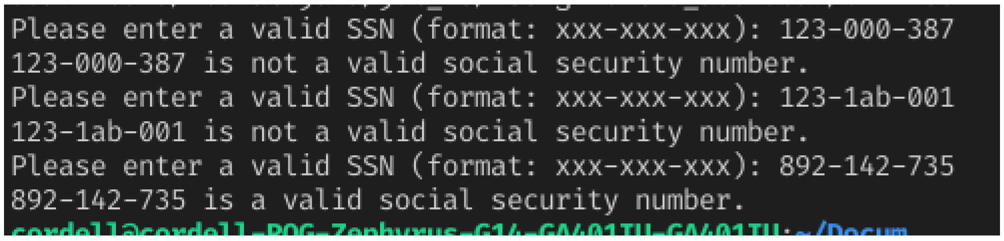
Filename: SSN.java

This program reads a string from the user in social security number format, then checks if the number is formatted correctly/valid. If the string is not formatted correctly, the user is prompted to re-enter the string. If the string is formatted correctly, a message is printed to the console.

Pseudocode:

```
START
PRINT "Please enter a social security number (format: xxx-xxx-xxx): "
READ socialSecurity
COMPUTE boolean isValid
IF socialSecurity length DOES NOT EQUAL 11
        PRINT socialSecurity " is not a valid social security number"
        COMPUTE isValid EQUALS false
ELSE IF socialSecurity at index 0 IS LESS THEN 1
        PRINT socialSecurity " is not a valid social security number"
        COMPUTE isValid EQUALS false
ELSE IF socialSecurity at index 4 IS LESS THEN 1
        PRINT socialSecurity " is not a valid social security number"
        COMPUTE isValid EQUALS false
ELSE IF socialSecurity at index 3 and 7 ARE NOT EQUAL TO "-"
        PRINT socialSecurity " is not a valid social security number"
        COMPUTE isValid EQUALS false
ELSE
        FOR integer i = 0 AS LONG AS i < 10 THEN i++
                IF i DOES NOT EQUAL 3 or 7
                        IF socialSecurity at index i IS NOT an integer between 0 – 9 inclusive
                                PRINT socialSecurity " is not a valid social security number"
                                COMPUTE isValid EQUALS false
IF isValid EQUALS true
        PRINT socialSecurity " is a valid social security number."
ELSE
        RESTART PROGRAM
END
```

Test Case:



```
Please enter a valid SSN (format: xxx-xxx-xxx): 123-000-387
123-000-387 is not a valid social security number.
Please enter a valid SSN (format: xxx-xxx-xxx): 123-1ab-001
123-1ab-001 is not a valid social security number.
Please enter a valid SSN (format: xxx-xxx-xxx): 892-142-735
892-142-735 is a valid social security number.
cordell@cordell-ROG-Zephyrus-G14-GA401IU-GA401IU:~/Docum
```

Palindrome Prime

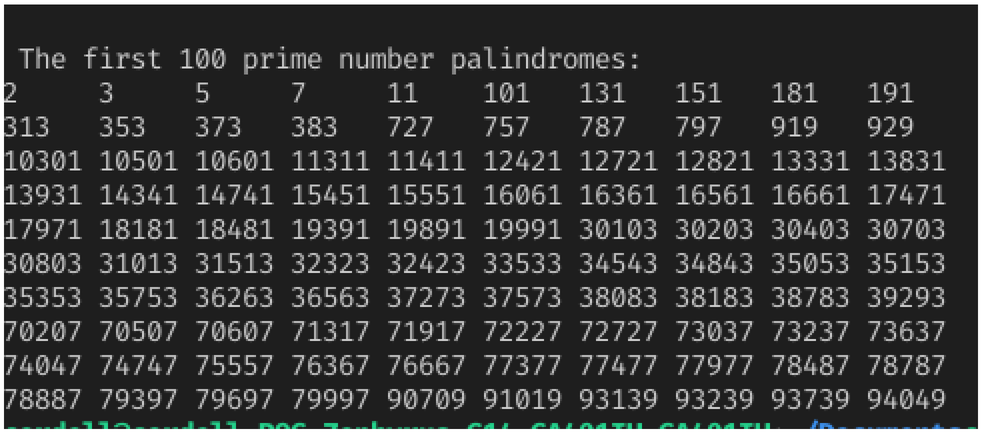
Filename: PalindromePrime.java

This program calculates the first 100 palindrome primes (upward from 0) and prints them in a 10x10 table in the console.

Pseudocode:

```
START
COMPUTE Integer Array primeNumbers
FOR Integer i equals 0 AS LONG AS i < 99999 THEN i++
    IF i EQUALS 1 or 0
        CONTINUE to next loop
    ELSE
        FOR Integer x equals 2 AS LONG AS x is less than or equal to i/2 THEN x++
            IF i % x is equal to 0
                CONTINUE to next loop
        COMPUTE add i to primeNumbers
COMPUTE Integer Array palindromePrimes
FOR Integer y equals 0 AS LONG AS y is less than length of primeNumbers THEN y++
    IF length of palindromePrimes equals 100
        BREAK the loop
    IF primeNumbers at index y IS LESS THAN 10
        COMPUTE add primeNumbers at index y to palindromePrimes
    COMPUTE String number equals value of primeNumbers a index y
    COMPUTE Integer length equals character length of number
    COMPUTE Integer half equals length divided by 2
    FOR Integer z equals 0 AS LONG AS z is less than half THEN z++
        IF character at index z of number IS EQUAL TO character at index length – 1 – z of number
            BREAK the loop
    COMPUTE add integer at index y from primeNumbers to palindromePrimes
COMPUTE Integer total = length of palindromePrimes
COMPUTE Constant Integer TOTAL= length of palindromePrimes
PRINT "The first 100 prime number palindromes:"
WHILE total IS GREATER THAN 0
    FOR Integer v  equals 0 AS LONG AS v is less than 10 THEN v++
        PRINT Formatted TOTAL – total-- (x10 per line, left justified)
END
```

Test Case:



```
 The first 100 prime number palindromes:
2     3     5     7     11    101   131   151   181   191
313   353   373   383   727   757   787   797   919   929
10301 10501 10601 11311 11411 12421 12721 12821 13331 13831
13931 14341 14741 15451 15551 16061 16361 16561 16661 17471
17971 18181 18481 19391 19891 19991 30103 30203 30403 30703
30803 31013 31513 32323 32423 33533 34543 34843 35053 35153
35353 35753 36263 36563 37273 37573 38083 38183 38783 39293
70207 70507 70607 71317 71917 72227 72727 73037 73237 73637
74047 74747 75557 76367 76667 77377 77477 77977 78487 78787
78887 79397 79697 79997 90709 91019 93139 93239 93739 94049
```

RSPGame

Filename: RSPGame.java

The third program is a rock paper scissors game. The user is asked to input one of three integers:

- 0 for rock
- 1 for scissors
- 2 for paper

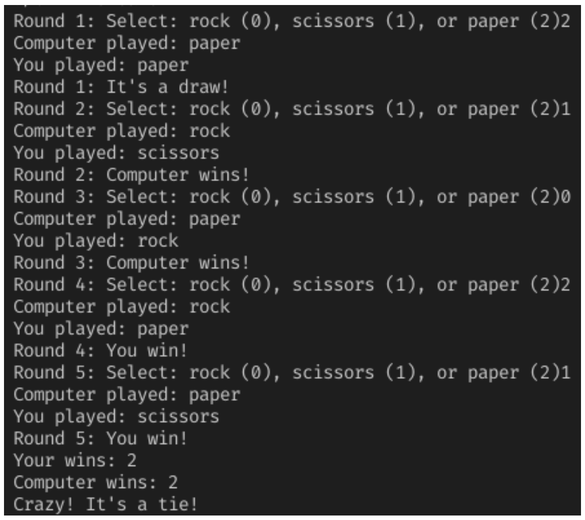
The program then randomly selects one of the three same integers. The two are then compared to determine if the computer or user wins or if there is a draw. If the player wins, their score increments likewise for the computer, though if there is a draw no points will be added. After 5 rounds the scores are compared, and a winner is announced in the console.

Pseudocode:

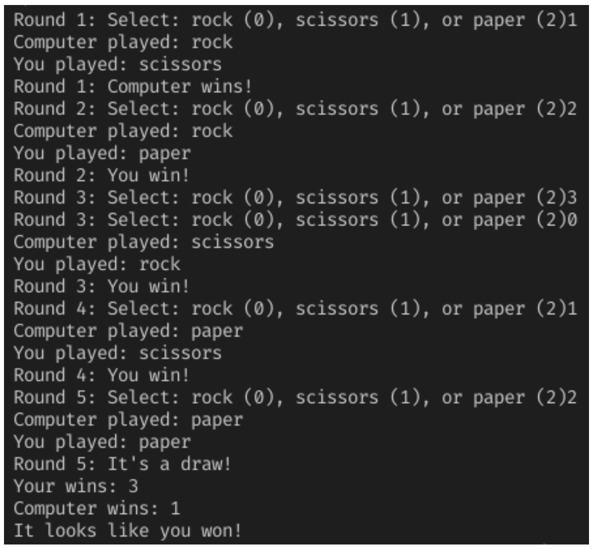
```
START
COMPUTE Integers playerScore = 0, compScore = 0, round = 1
WHILE round is less than or equal to 5
    PRINT ”Select: rock (0), scissors(1), paper(2)”
    READ int to user
    IF user > 2 OR user < 0
        RESTART LOOP
    COMPUTE int computer (random number between and including 0 and 2)
    PRINT “The computer played:” computer
```

```
        PRINT "You played:" user
        IF user = 0 and computer = 1 OR user = 1 and computer = 2 OR user = 2 and computer = 0
            PRINT "You win!"
            COMPUTE playerScore++
        ELSE IF computer = 0 and user = 1 OR computer = 1 and user = 2 OR computer = 2 and user = 0
            PRINT "You Loose!"
            COMPUTE compScore++
        ELSE
            PRINT "It's a draw!"
        COMPUTE round++
    PRINT "Your wins:" playerScore
    PRINT "Computer wins:" compScore
IF playerScore > compScore
    PRINT "You win!"
ELSE IF compScore > playerScore
    PRINT "Computer wins, you loose!"
ELSE
    PRINT "It's a tie!"
END
```

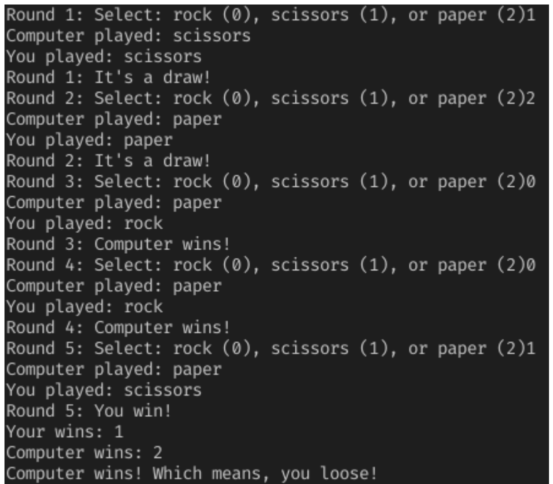
Test Cases:



```
Round 1: Select: rock (0), scissors (1), or paper (2)2
Computer played: paper
You played: paper
Round 1: It's a draw!
Round 2: Select: rock (0), scissors (1), or paper (2)1
Computer played: rock
You played: scissors
Round 2: Computer wins!
Round 3: Select: rock (0), scissors (1), or paper (2)0
Computer played: paper
You played: rock
Round 3: Computer wins!
Round 4: Select: rock (0), scissors (1), or paper (2)2
Computer played: rock
You played: paper
Round 4: You win!
Round 5: Select: rock (0), scissors (1), or paper (2)1
Computer played: paper
You played: scissors
Round 5: You win!
Your wins: 2
Computer wins: 2
Crazy! It's a tie!
```



```
Round 1: Select: rock (0), scissors (1), or paper (2)1
Computer played: rock
You played: scissors
Round 1: Computer wins!
Round 2: Select: rock (0), scissors (1), or paper (2)2
Computer played: rock
You played: paper
Round 2: You win!
Round 3: Select: rock (0), scissors (1), or paper (2)3
Round 3: Select: rock (0), scissors (1), or paper (2)0
Computer played: scissors
You played: rock
Round 3: You win!
Round 4: Select: rock (0), scissors (1), or paper (2)1
Computer played: paper
You played: scissors
Round 4: You win!
Round 5: Select: rock (0), scissors (1), or paper (2)2
Computer played: paper
You played: paper
Round 5: It's a draw!
Your wins: 3
Computer wins: 1
It looks like you won!
```



```
Round 1: Select: rock (0), scissors (1), or paper (2)1
Computer played: scissors
You played: scissors
Round 1: It's a draw!
Round 2: Select: rock (0), scissors (1), or paper (2)2
Computer played: paper
You played: paper
Round 2: It's a draw!
Round 3: Select: rock (0), scissors (1), or paper (2)0
Computer played: paper
You played: rock
Round 3: Computer wins!
Round 4: Select: rock (0), scissors (1), or paper (2)0
Computer played: paper
You played: rock
Round 4: Computer wins!
Round 5: Select: rock (0), scissors (1), or paper (2)1
Computer played: paper
You played: scissors
Round 5: You win!
Your wins: 1
Computer wins: 2
Computer wins! Which means, you loose!
```