Higher Diploma in Science in Computing

Software Development

Lecturer: Enda Stafford

Shan Liang X22187804

1. Digits mapped to the options/requirements

Student ID: X22187804

0 – AR1/AIB

4 – MAA1

0 – PWD2

2. IPO

2.1 Question 1

Input:

- Ask the number of aliases the user wants to validate.
- Ask the user to enter alias one by one.

Process:

- Creates a Scanner object to read user input.
- Create an instantiable class ValidPass.
- Covert aliases to lowercase and set them to ValidPass class.
- Call the validateAlias() method of the ValidPass class to validate the alias based on various conditions required including:
  - validate the alias length - 24 characters
  - validate the first 3 characters – aib
  - validate the rest of characters - digits from 0-9
  - validate the fixed characters - AIBxx**49210955**ddddddddcvn
  - validate the last 3 digits - in ascending order
- The validateAlias() method returns a boolean value indicating the validity of the alias.

Output:

- For each alias, display a message showing if the alias is valid.

2.2 Question 2

Input:

- Ask the user the number of passwords they want to generate.
- Ask the user to enter a number between 11 and 19 (inclusive) for each password.

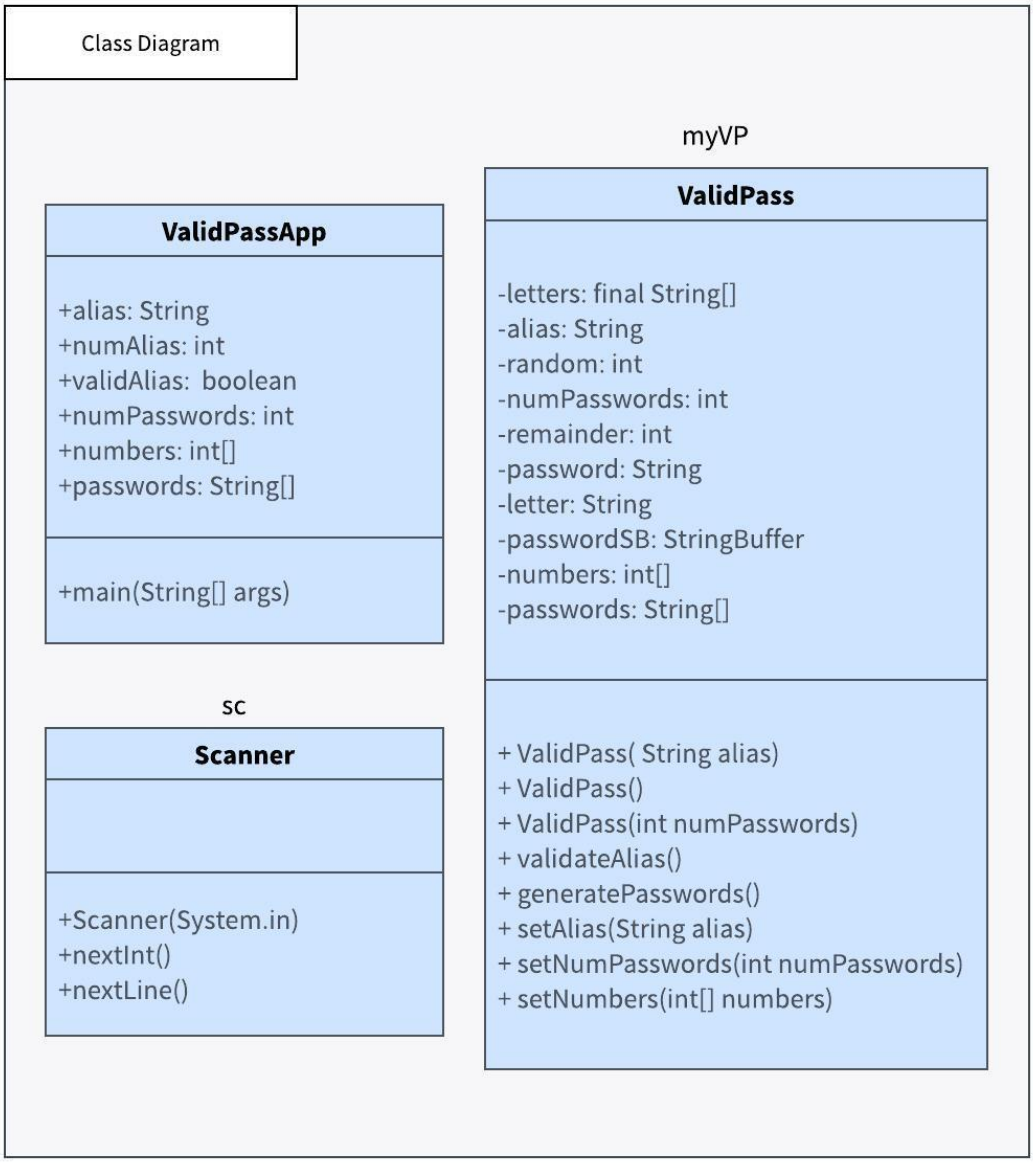
Process:

- Validate the number of passwords is at least 3.
- Validate each number entered for password are between 11 and 19 (inclusive).
- Set NumPasswords and Numbers to ValidPass class.
- Call generatePasswords() method in ValidPass class to generate passwords by:
  - randomly selecting 6 lowercase letters
  - adding an '#' symbol
  - calculating a remainder based on the fixed number 49556891
  - repeating the above process NumPasswords times
- The generatePasswords() method returns an array storing generated passwords.

Output:

- Display the generated passwords, one password per line.

3. Class diagram



4. Testing - screenshots of the output

4.1 Question 1

```
How many aliases would you like to validate?
10
Enter alias 1:
iiB004921095512345678257
The alias "iib004921095512345678257" is NOT VALID
Enter alias 2:
aiB0j4921095512345678257
The alias "aib0j4921095512345678257" is NOT VALID
Enter alias 3:
aiB004921895512345678257
The alias "aib004921895512345678257" is NOT VALID
Enter alias 4:
aiB00492109551234k678257
The alias "aib00492109551234k678257" is NOT VALID
Enter alias 5:
aiB004921095512345678253
The alias "aib004921095512345678253" is NOT VALID
Enter alias 6:
aiB004921095512345678257
The alias "aib004921095512345678257" is VALID
Enter alias 7:
AiB004921095565271283578
The alias "aib004921095565271283578" is VALID
Enter alias 8:
aiB004921095512345678253
The alias "aib004921095512345678253" is NOT VALID
Enter alias 9:
BiB004921095512345678253
The alias "bib004921095512345678253" is NOT VALID
Enter alias 10:
aiB00421095512345678253
The alias "aib00421095512345678253" is NOT VALID
```

Alias 1: when the first 3 characters are not “aib”.

Alias 2: when the following 2 characters are not digits from 0-9.

Alias 3: when the next 8 characters are not the same with “49210955”

Alias 4: when not all of the next 8 characters are digits from 0-9.

Alias 5: when the last 3 digits are not in an ascending order

Alias 6-10 are examples from the question 1 requirement form.

4.2 Question 2

```
Please enter the number of passwords to generate (minimum 3): 2
Number of passwords should be minimum 3. Please try again.
Please enter the number of passwords to generate (minimum 3):
9
Enter a number between 11 and 19 (inclusive) for password 1: 10
Number should be between 11 and 19 (inclusive). Please try again.
Enter a number between 11 and 19 (inclusive) for password 1: 20
Number should be between 11 and 19 (inclusive). Please try again.
Enter a number between 11 and 19 (inclusive) for password 1: 11
Enter a number between 11 and 19 (inclusive) for password 2: 12
Enter a number between 11 and 19 (inclusive) for password 3: 13
Enter a number between 11 and 19 (inclusive) for password 4: 14
Enter a number between 11 and 19 (inclusive) for password 5: 15
Enter a number between 11 and 19 (inclusive) for password 6: 16
Enter a number between 11 and 19 (inclusive) for password 7: 17
Enter a number between 11 and 19 (inclusive) for password 8: 18
Enter a number between 11 and 19 (inclusive) for password 9: 19
Generated passwords:
Password 1: qmormv#10
Password 2: ebsugd#11
Password 3: sdwerj#7
Password 4: ohvdvi#13
Password 5: ekhnux#11
Password 6: wiubwq#11
Password 7: tcfdwp#4
Password 8: rwjmdi#11
Password 9: xpryed#8
------------------------------------------------------------------------
BUILD SUCCESS
------------------------------------------------------------------------
```

Appendix:

```
/**
 * ValidPassApp.java
 * @author Shan Liang
 * 11/05/2023
 */

import java.util.Scanner;
public class ValidPassApp {

    public static void main(String[] args) {
        //declare variables - validate alias
        String alias;
        int numAliases;
        boolean validAlias;

        //declare variables - generate passwords
        int numPasswords;
        int[] numbers = new int[50];
        String[] passwords = new String[50];

        //declare and create objects
        Scanner sc = new Scanner(System.in);
        ValidPass myVP = new ValidPass();

        // validate alias
        //input
        System.out.println("How many aliases would you like to validate? ");
        numAliases = sc.nextInt();
```

```
        sc.nextLine(); //clear the line break

        for (int i = 1; i <= numAliases; i++) {
            System.out.println("Enter alias " + i + ": ");
            alias = sc.nextLine();
            alias = alias.toLowerCase();
            myVP.setAlias(alias);

            //compute
            validAlias = myVP.validateAlias();

            //output
            if(validAlias){
                System.out.println("The alias \"" + alias + "\" is VALID");
            }else{
                System.out.println("The alias \"" + alias + "\" is NOT VALID");
            }
        }

        //generate passwords
        //input - ask for the number of passwords
        System.out.print("Please enter the number of passwords to generate (minimum 3): ");
        numPasswords = sc.nextInt();

        // input validation
        while (numPasswords < 3) {
            System.out.println("Number of passwords should be minimum 3. Please try again.");
            System.out.println("Please enter the number of passwords to generate (minimum 3):
");
            numPasswords = sc.nextInt();
```

```
        }

        //send numPasswords to instantiable class
        myVP.setNumPasswords(numPasswords);

        //input - ask for the numbers between 11 and 19 (inclusive)
        for (int i = 0; i < numPasswords; i++) {
            System.out.print("Enter a number between 11 and 19 (inclusive) for password " + (i
+ 1) + ": ");
            numbers[i] = sc.nextInt();

            // Input validation
            while (numbers[i] < 11 || numbers[i] > 19) {
                System.out.println("Number should be between 11 and 19 (inclusive). Please
try again.");
                System.out.print("Enter a number between 11 and 19 (inclusive) for password
" + (i + 1) + ": ");
                numbers[i] = sc.nextInt();
            }
        }

        myVP.setNumbers(numbers);

        //generate passwords
        passwords = myVP.generatePasswords();

        //output
        System.out.println("Generated passwords:");
        for (int i = 0; i < numPasswords; i++) {
            System.out.println("Password " + (i + 1) + ": " + passwords[i]);
```

```
            }
        }
}



/**
  * ValidPass.java
  * @author Shan Liang
  * 11/05/2023
  */
public class ValidPass {
        //data members - validate alias
        private String alias;

        //data members - generate passwords
        private final String[] letters = {"a", "b", "c", "d", "e", "f", "g", "h", "i", "j", "k",
                "l", "m", "n", "o", "p", "q", "r", "s", "t", "u", "v", "w", "x", "y", "z"};
        private int random, numPasswords, remainder;
        private String password, letter;
        private StringBuffer passwordSB = new StringBuffer();
        private int[] numbers = new int[50];
        private String[] passwords = new String[50];

        //constructors
        public ValidPass(String alias) {
                this.alias = alias;
        }

        public ValidPass() {
        }
```

```
public ValidPass(int numPasswords) {
	this.numPasswords = numPasswords;
}



//compute methods
public boolean validateAlias(){
	//validate the alias length - 24 characters
	if(alias.length() != 24){
		return false;
	}

	//validate the first 3 characters - aib
	if(alias.charAt(0) != 'a'){
		return false;
	}
	if(alias.charAt(1) != 'i'){
		return false;
	}
	if(alias.charAt(2) != 'b'){
		return false;
	}

	//validate the rest of characters - digits from 0-9
	for (int i = 3; i < alias.length(); i++) {
		if(alias.charAt(i) != '0' && alias.charAt(i) != '1' &&
				alias.charAt(i) != '2' && alias.charAt(i) != '3' &&
				alias.charAt(i) != '4' && alias.charAt(i) != '5' &&
				alias.charAt(i) != '6' && alias.charAt(i) != '7' &&
```

```
                    alias.charAt(i) != '8' && alias.charAt(i) != '9'){
                return false;
            }
        }

        //validate the fixed characters - AIBxx49210955ddddddddcvn
        if(alias.charAt(5) != '4' || alias.charAt(6) != '9' ||
                alias.charAt(7) != '2' || alias.charAt(8) != '1' ||
                alias.charAt(9) != '0' || alias.charAt(10) != '9' ||
                alias.charAt(11) != '5' || alias.charAt(12) != '5'){
            return false;
        }

        //validate the last 3 digits - in ascending order
        for (int i = 21; i < 23; i++) {
            if((int)alias.charAt(i) >= (int)alias.charAt(i + 1)){
                return false;
            }
        }

        return true;
    }

    public String[] generatePasswords(){
        for (int i = 0; i < numPasswords; i++){
            //clear the StringBuffer
            passwordSB.setLength(0);

            // Generate 6 random letters
            for (int j = 0; j < 6; j++) {
```

```
                //generate a random number between 0-25
                random = (int) (Math.random() * (letters.length - 1));
                //generate a random letter
                letter = letters[random];
                passwordSB.append(letter);
            }

            //add an "#" symbol
            passwordSB.append("#");

            //calculate the remainder and add it on - 49556891
            remainder = 49556891 % numbers[i];
            passwordSB.append(remainder);

            //change the StringBuffer to String and add it to the array
            password = passwordSB.toString();
            passwords[i] = password;
        }
        return passwords;
    }

    //setters and getters
    public void setAlias(String alias) {
        this.alias = alias;
    }

    public void setNumPasswords(int numPasswords) {
        this.numPasswords = numPasswords;
    }
```

```
    public void setNumbers(int[] numbers) {
        this.numbers = numbers;
    }

}
```