CT2109: Assignment 1

Problem Statement

The problem consists of a creating a program that reads in the letters of the alphabet from the user console, one at a time. The user should be able to press enter between each letter and any incorrect letters or words should be ignored (i.e., the input is incorrect). If the user can input the alphabet in the correct sequence, whether it be backwards or forwards, the program should be able tell how long it took to complete the sequence.

Analysis

From the problem statement above, there are several key aspects that program should be able to handle. First, the program should ask the user whether the alphabet should be forwards or backwards. From that, the application should be able to populate an array containing the alphabet. As we know the length of the array in advance, we can use a basic array type, rather than a ArrayList, for example. The alphabet can be populated using ASCII values (a = 97, z = 122, etc.). Then a loop the length of the alphabet, could instantiate a Scanner that takes the value of each line the user enters and compare that to the current 'correct' letter that is in the array. An error message can be displayed in the console if the letter is incorrect. Also, we know that the correct input will be in the form of only one letter so that simplifies complexity regarding the validation of the input strings. To measure the time, the program can take the timestamp from the time at the end of the function subtracted from the time at the start of the function.

Design Notes

Getting the time elapsed in seconds.

```
double start = System.nanoTime();

double total = (System.nanoTime() - start) / 1_000_000_000.0;
```

Filling the alphabet array using ASCII values.

```
for (int i = 97; i < 123; i++) {

        a[i - 97] = (char)i;

}
```

After doing further research, I decided to fill the alphabet array using toCharArray().

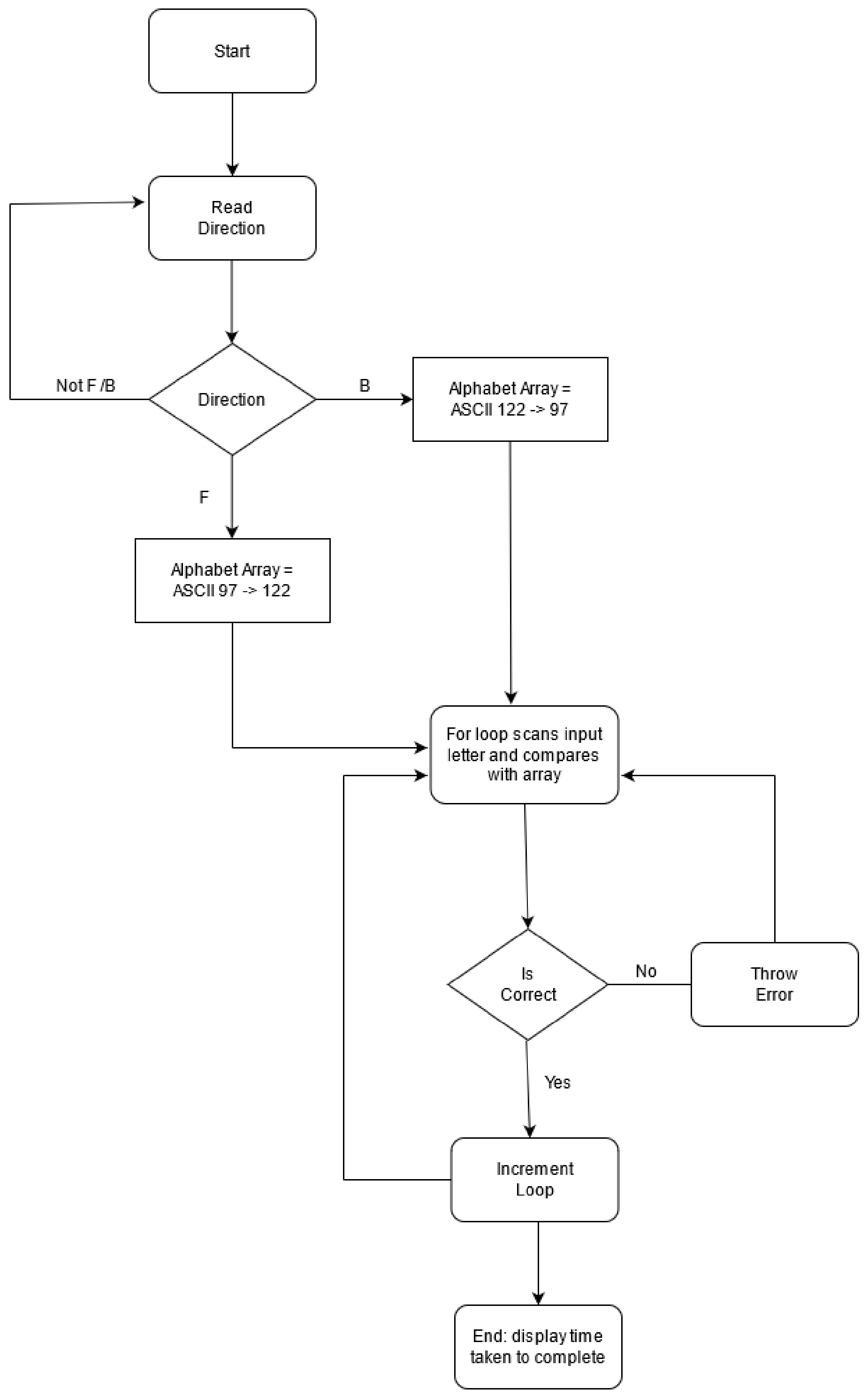
```
a = "abcdefghijklmnopqrstuvwxyz".toCharArray();

a = "zyxwvutsrqponmlkjihgfedcba".toCharArray();
```

A function processInput() will check if the input is a single letter. If it is, then it will be converted to lower case. If not, we can just run the function again until the user inputs something of length one. Thus, the program only needs to call this function to get an input value from the user.

A function named, initializeAlphabet() will create the alphabet array dependant on the direction the user specifies.

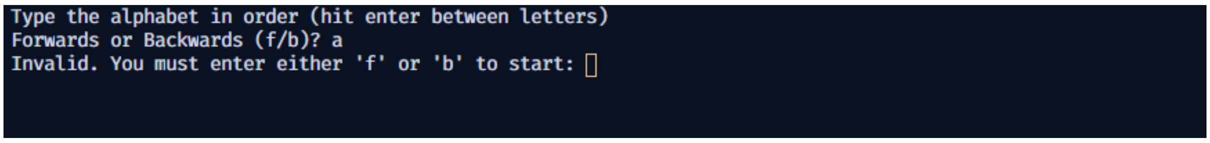
A basic flowchart demonstrates the flow for inputting the characters.


Start
Read Direction
Direction
Not F /B
B
F
Alphabet Array = ASCII 122 -> 97
Alphabet Array = ASCII 97 -> 122
For loop scans input letter and compares with array
Is Correct
No
Throw Error
Yes
Increment Loop
End: display time taken to complete

(The final code is further explained in the source code comments.)

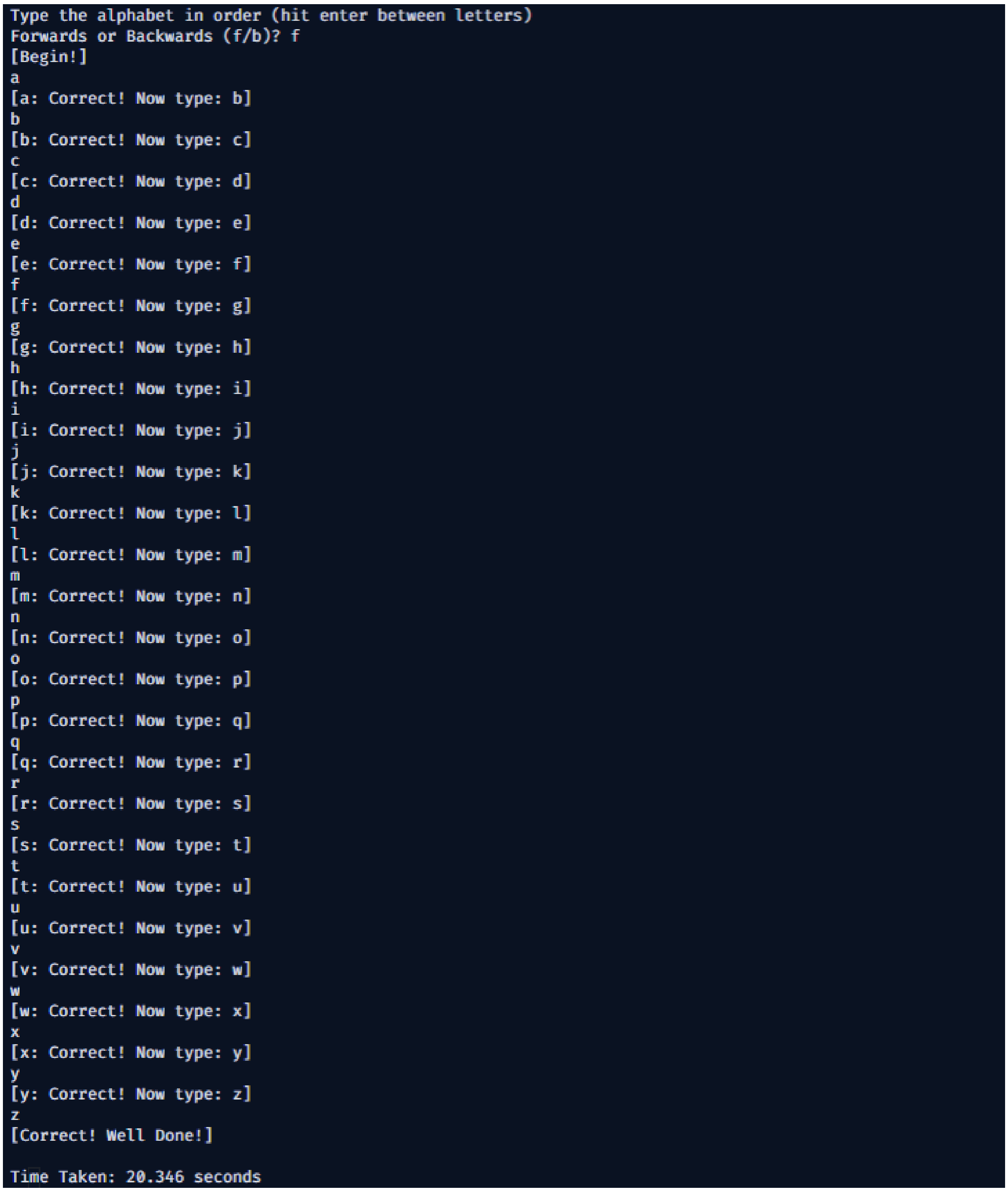
Testing

Case #1: Invalid direction specified



```
Type the alphabet in order (hit enter between letters)
Forwards or Backwards (f/b)? a
Invalid. You must enter either 'f' or 'b' to start: 
```

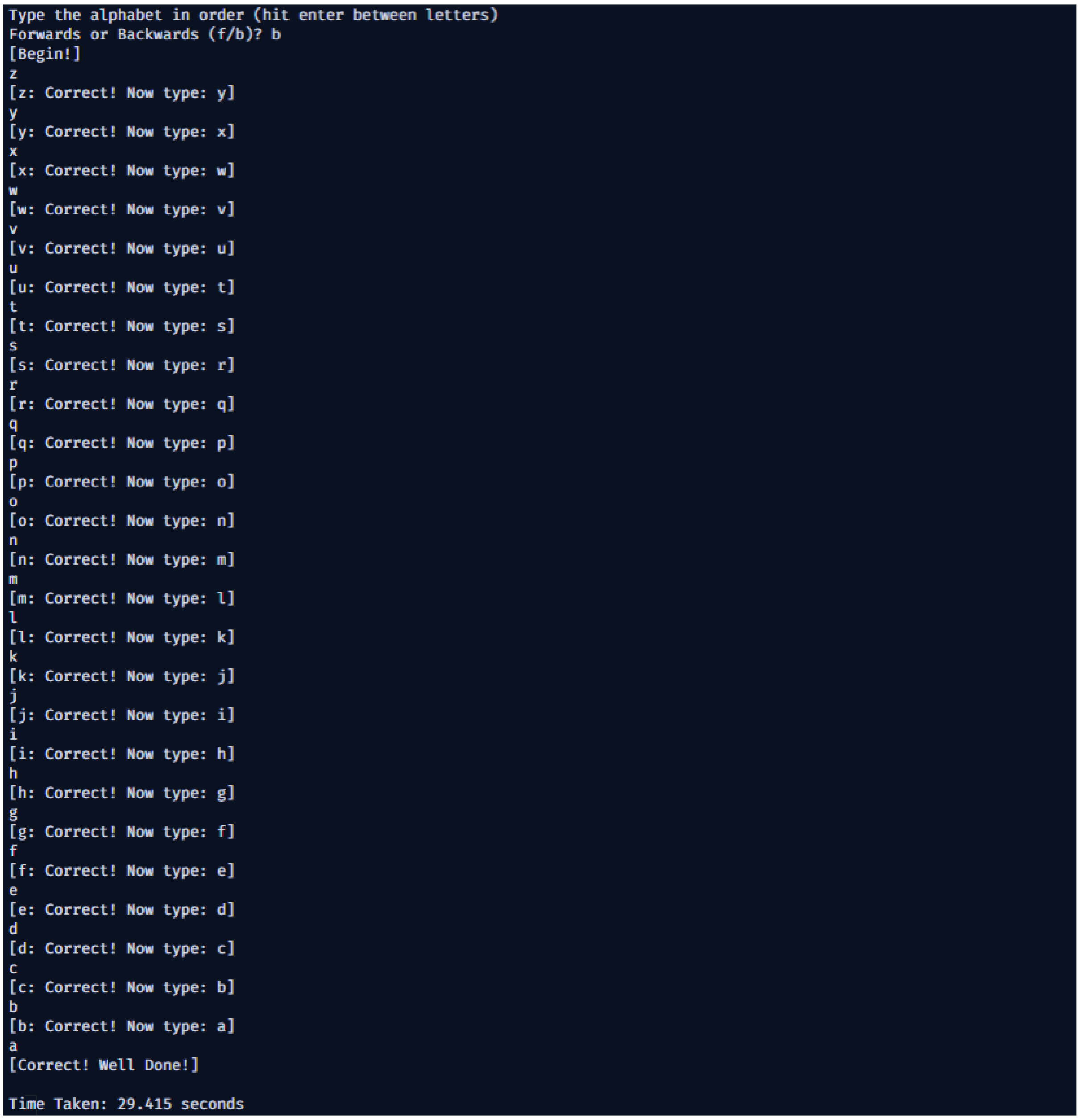
Case #2: Forwards is chosen



```
Type the alphabet in order (hit enter between letters)
Forwards or Backwards (f/b)? f
[Begin!]
a
[a: Correct! Now type: b]
b
[b: Correct! Now type: c]
c
[c: Correct! Now type: d]
d
[d: Correct! Now type: e]
e
[e: Correct! Now type: f]
f
[f: Correct! Now type: g]
g
[g: Correct! Now type: h]
h
[h: Correct! Now type: i]
i
[i: Correct! Now type: j]
j
[j: Correct! Now type: k]
k
[k: Correct! Now type: l]
l
[l: Correct! Now type: m]
m
[m: Correct! Now type: n]
n
[n: Correct! Now type: o]
o
[o: Correct! Now type: p]
p
[p: Correct! Now type: q]
q
[q: Correct! Now type: r]
r
[r: Correct! Now type: s]
s
[s: Correct! Now type: t]
t
[t: Correct! Now type: u]
u
[u: Correct! Now type: v]
v
[v: Correct! Now type: w]
w
[w: Correct! Now type: x]
x
[x: Correct! Now type: y]
y
[y: Correct! Now type: z]
z
[Correct! Well Done!]

Time Taken: 20.346 seconds
```

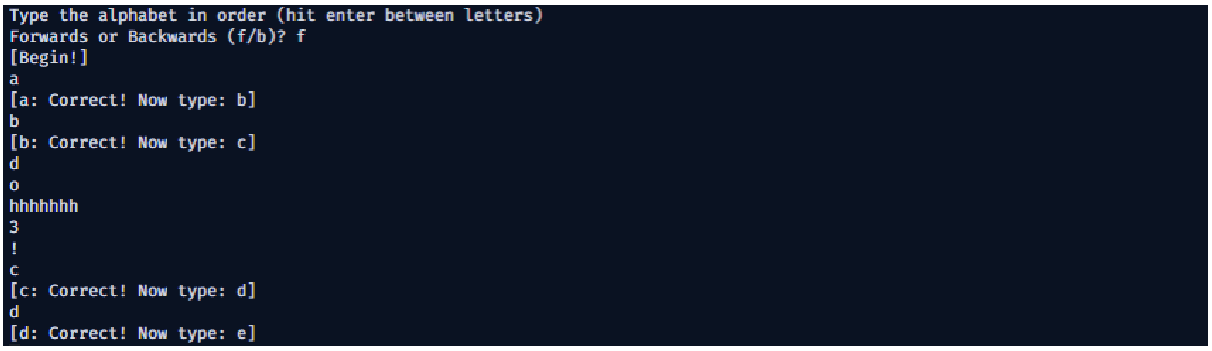
Case #3: Backwards is chosen



```
Type the alphabet in order (hit enter between letters)
Forwards or Backwards (f/b)? b
[Begin!]
z
[z: Correct! Now type: y]
y
[y: Correct! Now type: x]
x
[x: Correct! Now type: w]
w
[w: Correct! Now type: v]
v
[v: Correct! Now type: u]
u
[u: Correct! Now type: t]
t
[t: Correct! Now type: s]
s
[s: Correct! Now type: r]
r
[r: Correct! Now type: q]
q
[q: Correct! Now type: p]
p
[p: Correct! Now type: o]
o
[o: Correct! Now type: n]
n
[n: Correct! Now type: m]
m
[m: Correct! Now type: l]
l
[l: Correct! Now type: k]
k
[k: Correct! Now type: j]
j
[j: Correct! Now type: i]
i
[i: Correct! Now type: h]
h
[h: Correct! Now type: g]
g
[g: Correct! Now type: f]
f
[f: Correct! Now type: e]
e
[e: Correct! Now type: d]
d
[d: Correct! Now type: c]
c
[c: Correct! Now type: b]
b
[b: Correct! Now type: a]
a
[Correct! Well Done!]

Time Taken: 29.415 seconds
```

Case #4: Incorrect letter in the sequence



```
Type the alphabet in order (hit enter between letters)
Forwards or Backwards (f/b)? f
[Begin!]
a
[a: Correct! Now type: b]
b
[b: Correct! Now type: c]
d
o
hhhhhhh
3
!
c
[c: Correct! Now type: d]
d
[d: Correct! Now type: e]
```

Source Code

```
import java.util.Scanner;


public class Main {

    /**

     * This function contains the timer that will time how long it takes to input the 
alphabet.

     * For each letter in the alphabet array we'll get the input from the 
processInput() function and

     * check if it's correct. If it's correct, we'll move on to the next letter.

     */

    public static void main(String[] args) {

        System.out.print("Type the alphabet in order (hit enter between 
letters)\nForwards or Backwards (f/b)? ");


        // We use a normal array as we know what the size of the array is going to be 
beforehand.

        char alphabet[] = initializeAlphabet(), input;

        long start;


        System.out.printf("[Begin!]\n");


        // Create timer to monitor how long it takes to complete the alphabet.

        start = System.nanoTime();


        // Input and display each letter in the alphabet array

        for (int i = 0; i < alphabet.length; i++) {

            input = processInput();

            if (input == alphabet[i]) {

                // Check if end of array reached

                if (i == 25) {

                    System.out.print("[Correct! Well Done!]\n");

                } else {

                    System.out.printf("[%s: Correct! Now type: %s]\n", alphabet[i], 
alphabet[i + 1]);

                }

            } else {
                --i;
            }
        }

        System.out.printf("\nTime Taken: %.3f seconds", (System.nanoTime() - start) /
1_000_000_000.0);
    }

    /**
     * First we'll ask the user what direction they want the alphabet to be in. If
the user inputs F or B, we'll
     * create the array, else we'll just ask again by running this function again.
     */
    private static char[] initializeAlphabet() {
        char alphabet[] = new char[26], input;

        // First, we'll ask if the user wants the input forwards or backwards.
        input = processInput();

        // Initialize the alphabet array based on the what order the user has chosen.
        if (input == 'f') {
            alphabet = "abcdefghijklmnopqrstuvwxyz".toCharArray();
        } else if (input == 'b') {
            alphabet = "zyxwvutsrqponmlkjihgfedcba".toCharArray();
        } else {
            System.out.printf("Invalid. You must enter either 'f' or 'b' to start:
");
            return initializeAlphabet();
        }

        return alphabet;
    }

    /**
     * This function creates a scanner and takes in the user input, converting it to
lowercase and checking
     * the length to validate.
     */
    private static char processInput() {
        Scanner s = new Scanner(System.in);
        String line = s.nextLine();

        // If the length of the inputted string is 1, we'll return it so the main
function can check if it's correct.
        // To allow correct capital letters, we'll always return the lowercase of the
letter.
        if (line.length() == 1) {
            return line.toLowerCase().charAt(0);
        } else {
            return processInput();
        }
    }
}
```