CTU training solutions CAREER SUCCESS STARTS AT CTU

<table>
<tr><td colspan="10">Faculty of Information Technology</td></tr>
<tr><td rowspan="6">I declare that I am familiar with, and will abide to the Examination rules of CTU

 
Signature</td><td colspan="9">SUBJECT NAME: Beginner Java
SUBJECT CODE: J521</td></tr>
<tr><td colspan="4">FORMATIVE ASSESSMENT
Duration: Sep 15 – 16 Oct
Date 16 October 2023
Total Marks: 30
Total pages:</td><td colspan="5">Examiner: Faith Muwishi
Moderator: Mr. Newton</td></tr>
<tr><td colspan="9">Student number</td></tr>
<tr><td>2</td><td>0</td><td>2</td><td>3</td><td>2</td><td>7</td><td>5</td><td>9</td><td></td></tr>
<tr><td colspan="4">Surname:
Poponi</td><td colspan="3">Initials:
M.P</td><td>/</td><td>%</td></tr>
</table>

Table of Contents

Question Paper

A company has hired you to develop an application to generate passwords for their employees' accounts. The application prompts the user to provide one single line of text with an employee's full name in the following format "forename surname" at a time. (Please note that you are not required to validate the input, we assume that the input is well-formed.) Next, the application uses the given full name to create the corresponding item. Note that the application should work irrespective of how the user provides the full name i.e. using upper-case letters, lower-case letters, or a combination of both upper-case and lower-case letters. You should use the full name as provided i.e. you should not modify the given full name to a different letter case.

Assigned Item Rules to Create the Item Examples

(Given full name and corresponding item that should be created)

Password The password is created using the following rules:

• The letters 'a', 'e', and 't' from the given full name will not be used in the password

• Each vowel (except 'a' and 'e' which are eliminated) is going to be added twice

• Each space is replaced by the letter 'S' followed by a '&' and a '?'

• All the other characters will remain the same as in the given full name

• The password ends with the total number of letters eliminated (i.e. the total number of letters 'a', 'e', and 't' from the given full name that were not used in the password) For example, if the instantiable class receives:

• The full name "JANE dOe" then the compute method should create the password "JNS&?dOO3" (note that 3 is the total number of letters eliminated i.e. as the letters 'a' and 'e' from the given full name were not used in the password)

• The full name "Conor MUrphy" then the compute method should create the password "coonOOrS&?MUUrphy0" (note that 0 is the total number of letters eliminated i.e. as the given full name did not contain any of the letters 'a', 'e', or 't') Ask the user at the beginning of the application how many items they would like to create, and ensure that the application enables the user to provide that number of full names and for each full name create the corresponding password.

The source code should be commented throughout highlighting and explaining where the key functionality is being addressed.

Submission Format

1. Include a 5 minutes video where you demonstrate how the program is working the program should be visible in the video.
2. Submit a pdf document with all the program code together with the program output screenshot of the interface.

Source code:

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace PasswaordGenerator_Mzukisi_20232759
{
    internal class Program
    {
        // PasswordGenerator class
        class PasswordGenerator
        {
            // Declare the numItems variable
            private int numItems;

            // Get the number of items method
            public void GetNumItems()
            {
                // Ask the user for the number of items to create
                Console.Write("Enter the number of items to create: ");
                string input = Console.ReadLine();

                // Validate the input (validation)
                while (!int.TryParse(input, out numItems) || numItems <= 0)
                {
                    Console.WriteLine("\nInvalid input. Please enter a positive integer.");
                    Console.Write("Enter the number of items to create: ");
                    input = Console.ReadLine();
                }
            }

            // Generate password method
            public string GeneratePassword(string fullName)
            {
                // Declare variables to be used, like the list of characters for all the vowels
                // The password
                // Number of eliminated letters
                char[] vowels = { 'a', 'e', 'i', 'o', 'u', 't' };
```

```
            string password = "";
            int eliminatedLetters = 0;

            // Foreach loop to enumerate through the string and perform the appropriate
operations
            foreach (char c in fullName)
            {
                if (Array.IndexOf(vowels, Char.ToLower(c)) != -1)
                {
                    if (Char.ToLower(c) == 'a' || Char.ToLower(c) == 'e' || Char.ToLower(c) == 't')
                    {
                        eliminatedLetters++;
                    }
                    else
                    {
                        password += new string(c, 2);
                    }
                }
                else if (c == ' ')
                {
                    password += "S&?";
                }
                else
                {
                    password += c;
                }
            }

            password += eliminatedLetters;
            return password;
        }

        // Create Password method
        public void CreatePasswords()
        {
            // Get the number of items from the user (Call method)
            GetNumItems();

            // Create passwords for each item
            for (int i = 0; i < numItems; i++)
            {
                // Ask the user for the full name
                Console.Write("\nEnter the full name: ");
                string fullName = Console.ReadLine();

                // Generate the password
                string password = GeneratePassword(fullName);
                Console.WriteLine("Generated password: " + password);
            }
        }
    }
    static void Main(string[] args)
```

```
        {
            // Create an instance of the PasswordGenerator class
            PasswordGenerator generator = new PasswordGenerator();

            // Call the CreatePasswords method to start the password generation process
            generator.CreatePasswords();

            // Print an end program message
            Console.WriteLine("\nThank you for using the Password Generator, Goodbye!");

            // Wait for the user to enter a key before closing the application
            Console.ReadKey();
        }
    }
}
```

Output: