

**ELEC362 Assessment 1**

**Ben Hague (201146260)**

**Department of Electrical Engineering and Electronics**

**22 November 2018**

# Abstract

This Report outlines the 2 tasks given in ELEC362 Assignment 1 and discusses the solution to the problem and how the code has been constructed. We then move on to show the testing of the program and how it successfully achieves the specification.

# Declaration

**I confirm that I have read and understood the University’s definitions of plagiarism and collusion from the Code of Practice on Assessment. I confirm that I have neither committed plagiarism in the completion of this work nor have I colluded with any other party in the preparation and production of this work. The work presented here is my own and in my own words except where I have clearly indicated and acknowledged that I have quoted or used figures from published or unpublished sources (including the web). I understand the consequences of engaging in plagiarism and collusion as described in the Code of Practice on Assessment (Appendix L).**

Contents

[Abstract 1](#_Toc530659467)

[Declaration 1](#_Toc530659468)

[Part 1: A Non-linear Circuit Problem 3](#_Toc530659469)

[Introduction 3](#_Toc530659470)

[Program Design 3](#_Toc530659471)

[Results and Testing 4](#_Toc530659472)

[Obstacles 6](#_Toc530659473)

[Part 2: Programming Exercise 7](#_Toc530659474)

[Introduction 7](#_Toc530659475)

[Program Design 7](#_Toc530659476)

[Results and Testing 7](#_Toc530659477)

[Obstacles 9](#_Toc530659478)

[Appendix 10](#_Toc530659479)

[Appendix 1: Code for Part 1 10](#_Toc530659480)

[Appendix 2: Code for Part 2 14](#_Toc530659481)

# Part 1: A Non-linear Circuit Problem

## Introduction

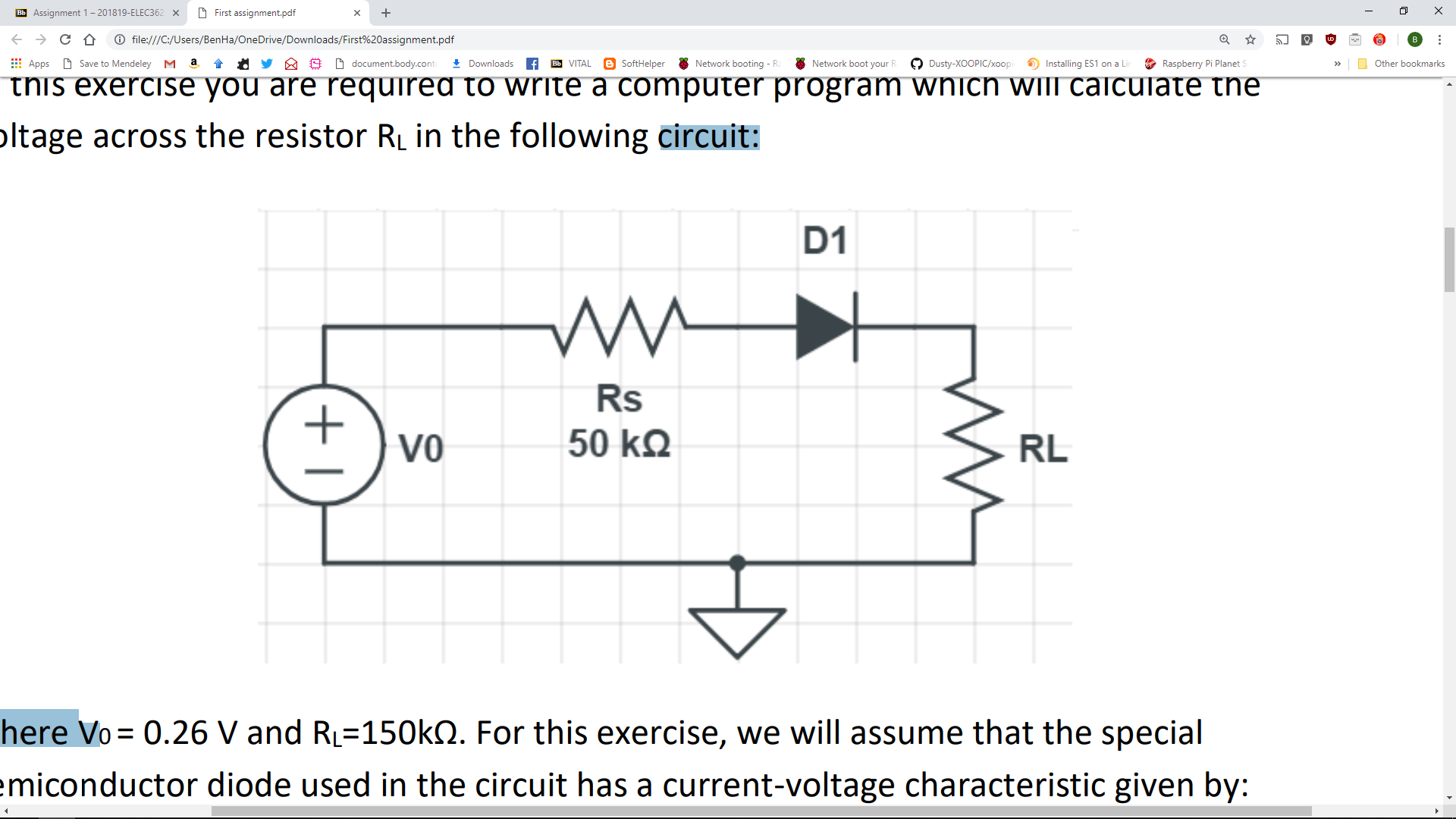
The first task is to write a program which will calculate the voltage drop across the Resistor R­L in the circuit diagram shown in Figure 1. For this process we will be making the assumption that V0 = 0.26v and RL­ = 150 kΩ. we are also going to take the assumption that the special semiconductor diode has a current voltage characteristic given by:

Figure 1: Circuit Diagram for Task 1

Where *i* is the diode current and VD the voltage across the diode. e/kT= 40 at room temperature and I = 8x10-8 .

## Program Design

The Program is designed with an object orientated approach. There are 4 object types:

* Diode
* Resistor
* Voltage Source
* Circuit

­The Program is designed such that each object is created, with its key attributes, each of these objects is given to the Circuit object during its creation. We can then call the voltage over the diode from within the circuit. From this we can then calculate the voltage drop over RL.

Below is the class diagram for each key component

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Diode |  | Resistor |  | VoltageSource |  | Circuit |
| private:  double EKT  public: |  | private:  double Resistance  public: |  | private:  double Current  double Voltage |  | private:  double RS  double RL  double V  double I  double EKT  double Precision = 0.0001 |
| private:  public:  Diode(double newEKT)  double getEKT() |  | private:  public:  Resistor(double R)  double getResistance() |  | private:  public:  VoltageSource(double V, double I)  double getCurrent()  double getVoltage() |  | private:  double FunctionOfX()  double FunctionOfXD()  public:  Circuit(VoltageSource Voltage, Resistor Rload, Resistor RSource, Diode Diode)  double CalculateVD(double Xi = 0.1)  double CalculateVRL()  void Summary() |

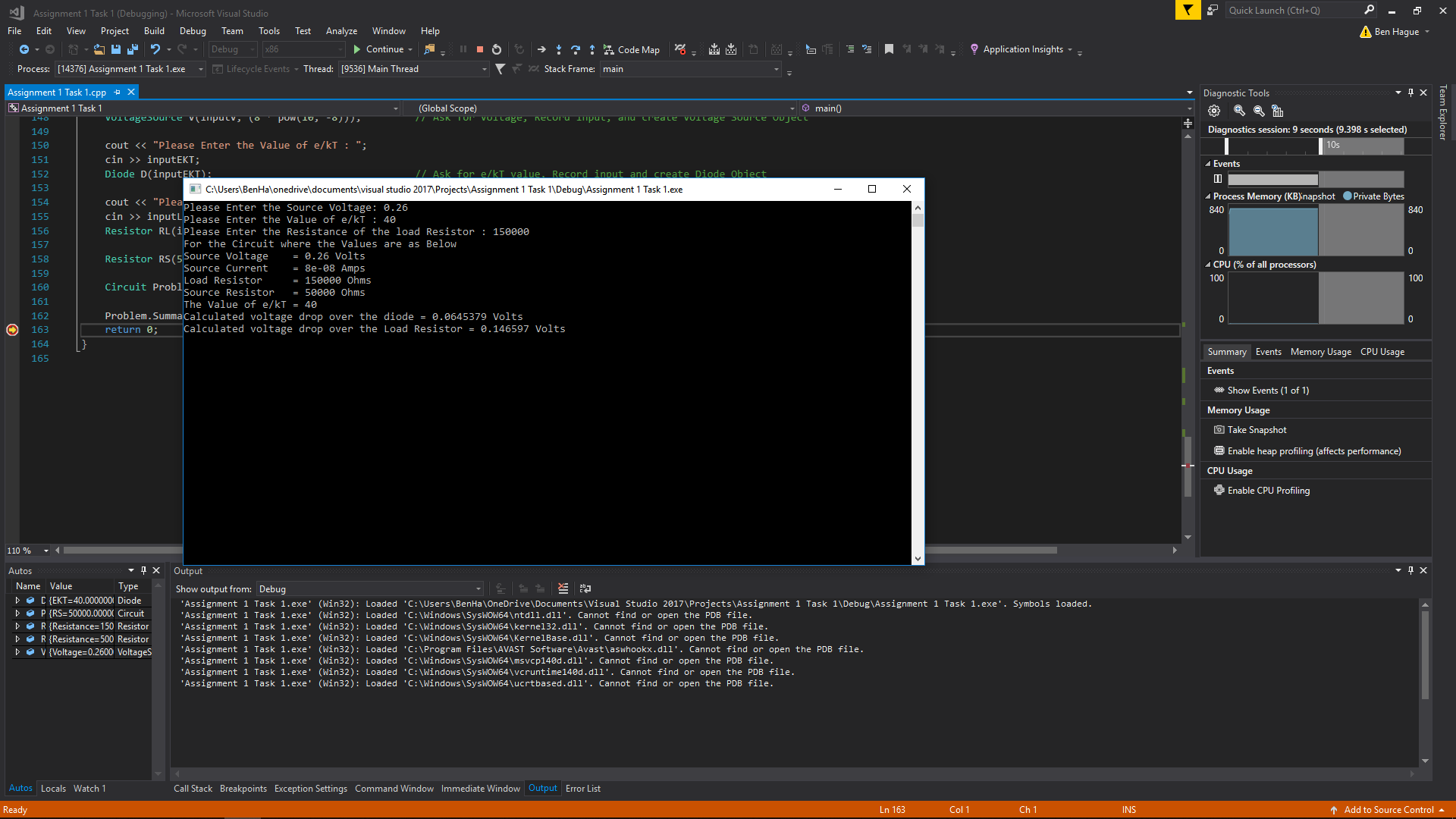
This structure to design the program allows a large amount of redundancy and scalability. This means that with minimal additional modification the program could be used to calculate multiple different styles of circuit.

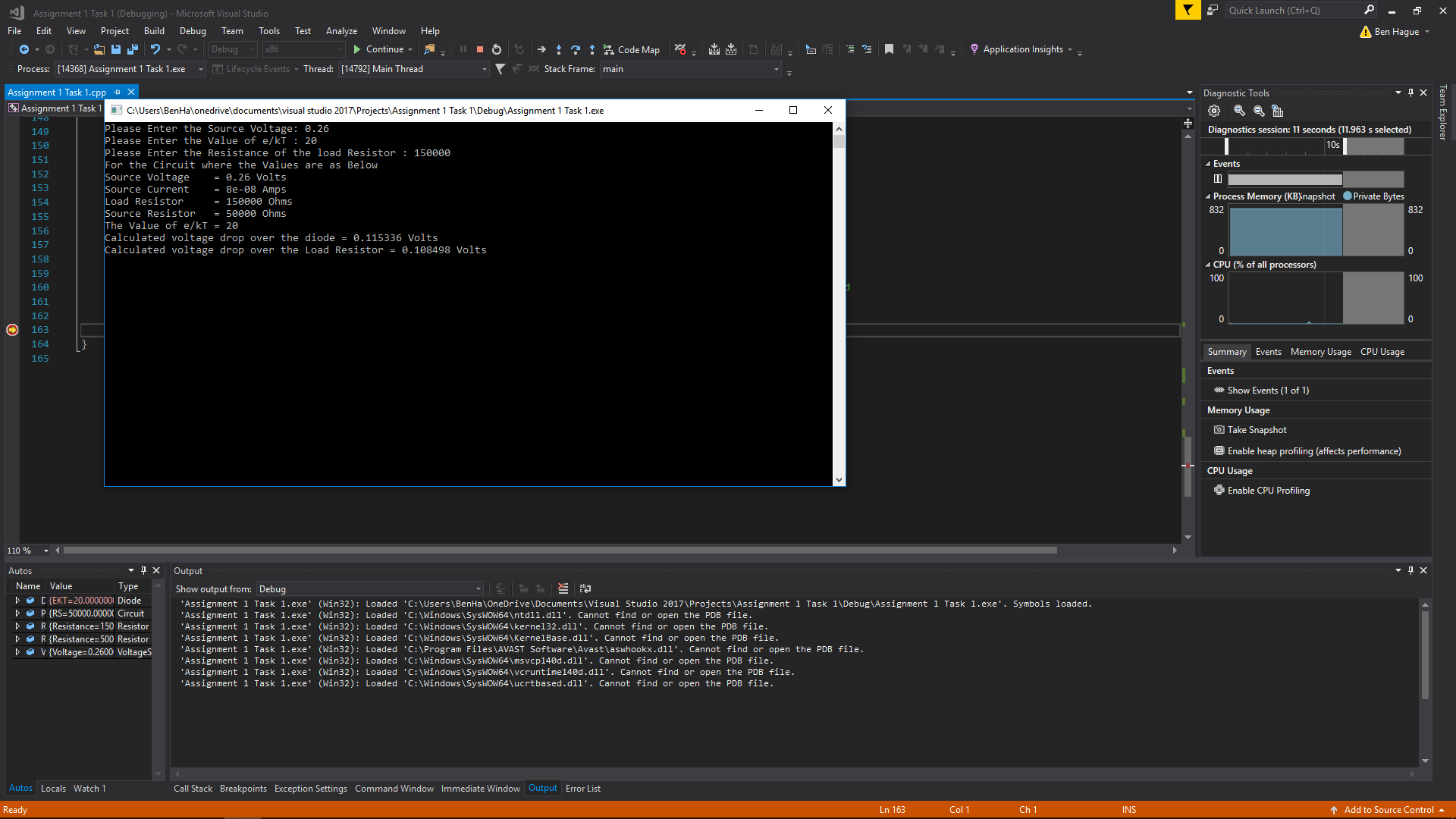
## Results and Testing

As Shown in the screenshots I have tested this with the results in Table 1 which details the output inputs and if the result was as expected

Table 1: Test vales for task 1

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| e/kT | RL | V0 | Voltage Over D | Voltage over RL | Comment | Screenshot |
| 40 | 150000 | 0.26 | 0.0645379 | 0.146597 | All as expected | Figure 2 |
| 20 | 150000 | 0.26 | 0.115336 | 0.108498 | All as expected | Figure 3 |

Figure 2: Screenshot from part 1 testing

Figure 3 Screenshot from part 1 testing

## Obstacles

The main obstacle with this code was to effectively code the newton Raphson method, I ended up deciding to use recursion as this allowed me to have effective code which would not endlessly loop. I also had an issue with the maths as this is not one of my strong points.

# Part 2: Programming Exercise

## Introduction

The second task of this assignment is to write a program which reads 5 full names from the keyboard, capitalises the first letter of each name and stores the names on a new line in a known file.

## Program Design

We can split the task into 3 different sections

Data Collection – the process of collecting the name and storing it in a variable

Data Processing – how we capitalise different names and putting them in a storable format

Data Storage – how we store the names in the file

For the data collection we would take the input and ensure it only uses valid characters and return the output.

For data processing we separate the first, last and middle names with commas. This means that they can be outputted to a comma separated value file and opened with a spreadsheet program.

The data storage stores the data in a specified file for later use.

This process is then repeated for the 5 different names.

## Results and Testing

The testing strategy is to input variety of inputs and record the output to show how they work.

Table 2: Test Values for task 2

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Name 1 | Name 2 | Name 3 | Name 4 | Name 5 | Results | Comment | Screenshot |
| Ben Hague | Ben hague | ben Hague | Ben H4gue | Ben o’hague | As expected | Wouldn’t accept name 4 as input due to the number 4 | Figure 4 |
| Ben hague-hague | Benjamin George Hague | Ben Georg3 Hague-kin | ben George harry hague | Ben Ha£ue | As expected | Rejected input on names 3 and 5 | Figure 5 |

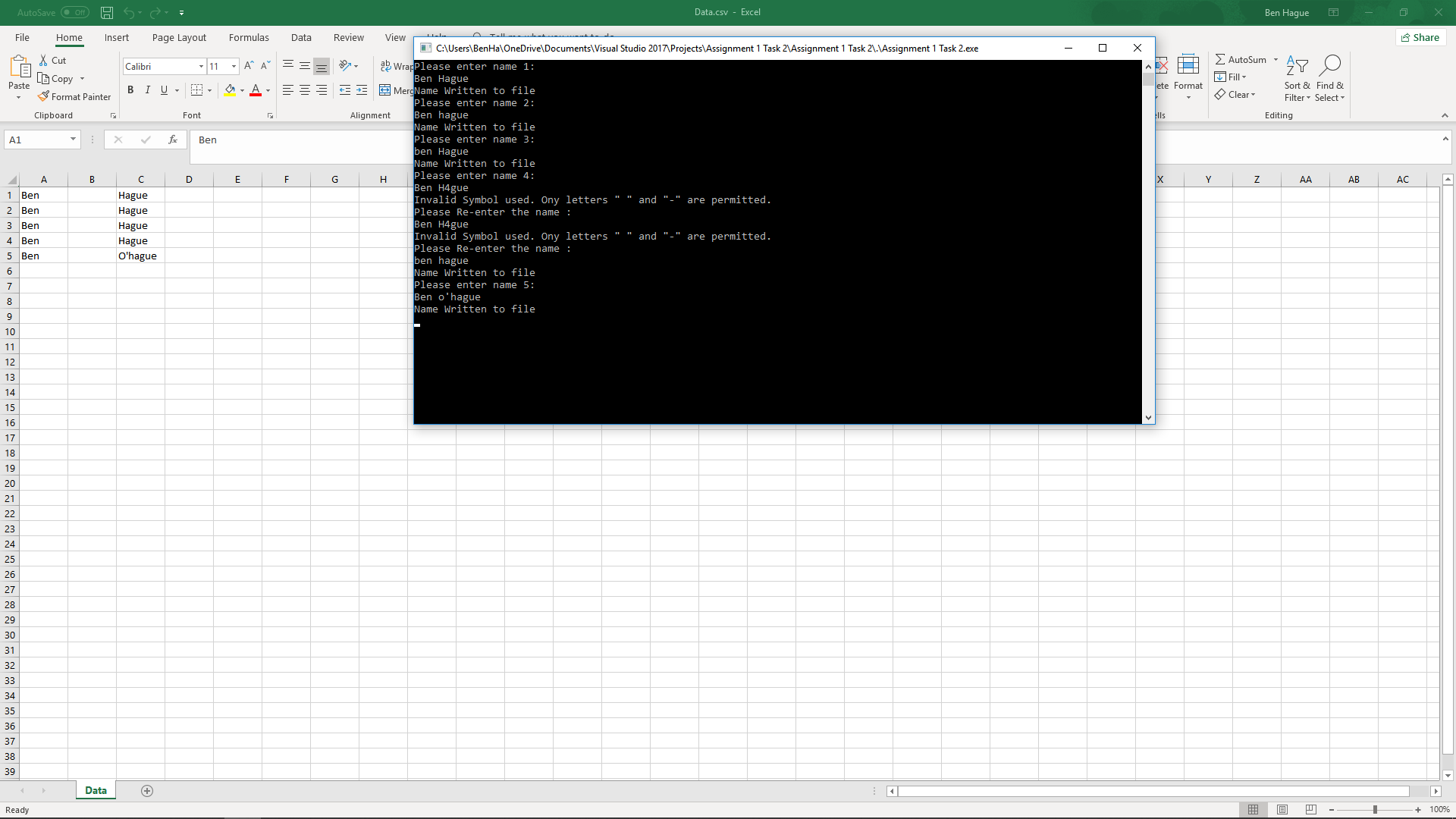


Figure 4: Screenshot from part 2 testing

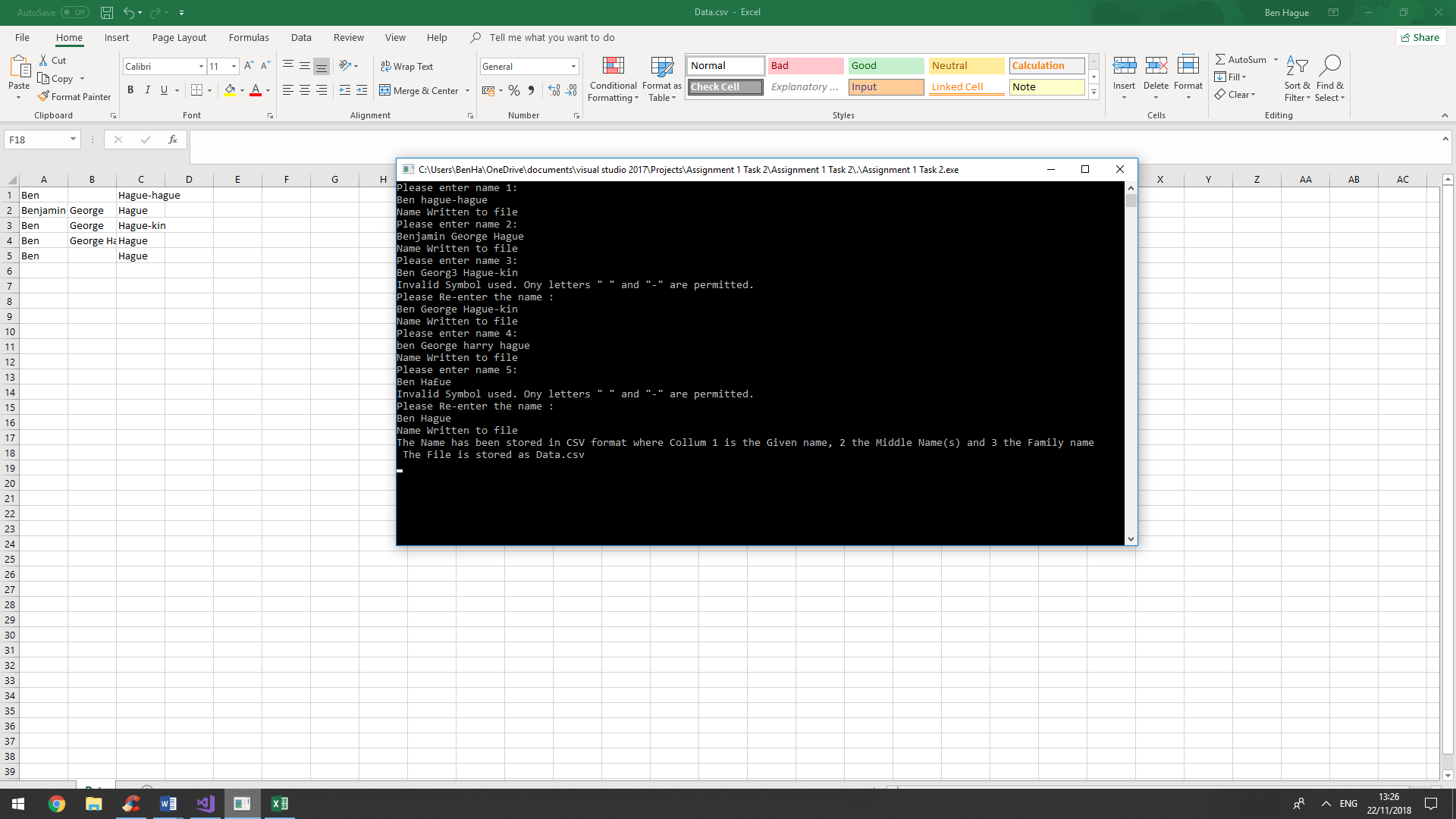


Figure 5: Screenshot from part 2 testing

## Obstacles

The “£” sign proved problematic and did not work with the to upper command in C, to avoid this issue I wrote a short Capitalise code which converted it. I also had to include separate code to negotiate spaces being present at the start or end as it caused an input issue.

# Appendix

## Appendix 1: Code for Part 1

// Assignment 1 Task 1.cpp

// Assignmnet 1 is a piece of code which calculates the Voltage drop over a diode in a simple circuit

// The assignment uses an Object Orientated approach for data management and calculations

#include "stdafx.h"

#include <cmath>

#include "iostream"

using namespace std;

/\*

The first section creates object types for the key 3 types of components in the circuit:

Diode holds the value for e/kT, this is a constant we use which is dependant on the diode.

Resistor holds a resistance value.

Voltage Source holds the current and voltage in the circuit.

\*/

class Diode

{

private:

double EKT; // Declare the double EKT in a private instance

public:

Diode(double newEKT) // Constructor, the diode object is created with the value for e/kT

{

EKT = newEKT;

}

double getEKT() // public Method returns the value of private Variable EKT for use in calculation.

{

return EKT;

}

};

class Resistor

{

private:

double Resistance; // Declare the double Resistance in a private instance.

public:

Resistor(double R)

{

Resistance = R; // Constructor, the resistor object is created with the value for Resistance

}

double getResistance()

{

return Resistance; // public Method returns the value of private Variable Resistance for use in calculation

}

};

class VoltageSource

{

private:

double Voltage, Current; // Declare the variables Voltage and current as doubles in a private instance

public:

VoltageSource(double V , double I) // Constructor, the VoltageSource object is created with a value for Voltage and current

{

Voltage = V;

Current = I;

}

double getVoltage() // public Method returns the value of private Variable Voltage for use in calcuation

{

return Voltage;

}

double getCurrent() // public Method returns the value of private Variable Current for use in calcuation

{

return Current;

}

};

/\*

The Second Section Contains the Circuit object

The Circuit object is given the relevent components and then we can use this to discover the voltage drop over the diode

\*/

class Circuit // Define a class Circuit for finding the Calculations

{

private:

double RS, RL, V, EKT,I; // Define doubles For Source Resistance,Load Resistance, Voltage, E/KT and Current

double Precision = 0.0001; // Declare the precicsion as 5 significant figures

double FunctionOfX(double x) // Calculates the Function of X (Inluded for completeness and modularity)

{

double Fx;

Fx = (I\*(exp(EKT\*x) - 1) - ((V - x) / (RS + RL)));

return Fx;

}

double FunctionOfXD(double x) // Calculates the derivative for the Function of X (Inluded for completeness and modularity)

{

double FDx;

FDx = I\*(EKT\*exp(EKT\*x)) + (1 / (RS + RL));

return FDx;

}

public:

Circuit(VoltageSource Voltage, Resistor Rload, Resistor RSource, Diode Diode) // Constructors takes input of a set of components (Voltage Source, 2 Resistors and a diode)

{

V = Voltage.getVoltage(); // Take the needed variables from each of the input components and store it in a relevant Local Variable

I = Voltage.getCurrent();

RS = RSource.getResistance();

RL = Rload.getResistance();

EKT = Diode.getEKT();

}

double CalculateVD(double Xi = 0.1) // Uses the Newton Raphson Method to calculate the voltage drop over the diode takes an input of a guessed preliminary x value

{

double Xii = (Xi - (FunctionOfX(Xi) / FunctionOfXD(Xi))); // Calculate the next value of X "Xii"

if (abs((Xii - Xi) / Xi) >= Precision) // Determine whether the new value of x is right for the current number of significant figures.

{

return CalculateVD(Xii); // If not right, return the value given by running The newton Raphson method on the current value of Xii

}

else {

return Xi; // If right return the value for Xi

}

}

double CalculateVRL() // Calculates the value of the voltage drop over RL

{

double VD = CalculateVD(); // Gets the Voltage Drop over the diode

return (V - VD)\*(RL / (RS + RL)); // Returns the voltage drop over the resistor

}

void Summary() // Summary prints a summary of the calculations to the console window.

{

cout << "For the Circuit where the Values are as Below" << endl;

cout << "Source Voltage = " << V << " Volts"<< endl;

cout << "Source Current = " << I << " Amps" << endl;

cout << "Load Resistor = " << RL << " Ohms" << endl;

cout << "Source Resistor = " << RS << " Ohms" << endl;

cout << "The Value of e/kT = " << EKT << endl;

cout << "Calculated voltage drop over the diode = "<< CalculateVD() << " Volts" << endl;

cout << "Calculated voltage drop over the Load Resistor = " << CalculateVRL() << " Volts" << endl;

}

};

/\*

The third part is the main class

This part handles user interaction, gets inputs, declares objects and does all the other bits and bobs

\*/

int main()

{

double inputV, inputEKT, inputLR; // Declare variables for the inputs

cout << "Please Enter the Source Voltage: ";

cin >> inputV;

VoltageSource V(inputV, (8 \* pow(10, -8))); // Ask for Voltage, Record input, and create Voltage Source Object

cout << "Please Enter the Value of e/kT : ";

cin >> inputEKT;

Diode D(inputEKT); // Ask for e/kT value, Record input and create Diode Object

cout << "Please Enter the Resistance of the load Resistor : ";

cin >> inputLR ;

Resistor RL(inputLR); // Ask for Load Resistance, Record input and create Diode Object

Resistor RS(50000); // Create Resistor Object with a resistance of 50000ohms

Circuit Problem( V, RL, RS, D); // Create Object for the circuit Giving the circuit the components needed

Problem.Summary(); // Print Summary of the Problem

return 0;

}

## Appendix 2: Code for Part 2

// Assignment 1 Task 2.cpp : Defines the entry point for the console application.

//

#include "stdafx.h"

#include "iostream"

#include <string>

#include <fstream>

using namespace std;

char Capitalise(char InputChar) // Capitalise is a short segment of code using the Ascii value of a letter to determine if it is lowercase and if so, capitalise it

{ // Takes a char input and gives a char output

int InputInt = (int)InputChar; // Find the Ascii Value for input char

if ('a' <= InputInt && InputInt <= 'z') // if the input char is lower case 97 is "a" and 122 is "z"

{

InputInt -= 32; // Convert the Input char to a capital by subtracting 32 from the char value (Ascii table magic)

}

return (char)InputInt; // return the new char

}

string PrepName(string Input) // SentanceCase is a function which looks for spaces within a statement and capitalises the first letter after the space useing Capitalise

{ // Takes a string input and gives a string output

Input[0] = Capitalise(Input[0]); // This line is important as it ensures the first letter of the string is capitalised

int Fspace = 1000; // this is used to find the first space, we choose an unrealisticly high number to allow the first space to be identified

int Lspace = 0; // this is used to find the last space, we make the first number 0 as this is the begininng of the array

for (int i = 0; i < (int)Input.length()-1; i++) // For i in the range of 0 to the length of the string

{

if ((char)Input[i] == ' ') // if the char is a space

{

if (i < Fspace) { Fspace = i; } // store the location for the first space in Fspace

if (i > Lspace) { Lspace = i; } // store the location for the last space in Lspace

Input[i + 1] = Capitalise(Input[i + 1]); // use the Capitalise function to ensure the next charachter is a capital

}

}

Input[Fspace] = ','; // Convert the first space to a comma for the csv file

if (Fspace == Lspace) { // if there are only two names determined by the difference between first and last space

Input.insert(Lspace, ","); // insert an additional comma to indicate a blank colum for middle name

}

else { // if there are not 2 names

Input[Lspace] = ','; // replace the final space to a comma at the end of the middle names

}

return Input; // Return the new modified comma deliminated string

}

string GetText() // Get the user input and verify it

{

string Input = " "; // Declare the input string

try

{

getline(cin, Input); // Get the input from the screen

for (int i = 0; i < ((int)Input.length() - 1); i++) // For i in the range of 0 to the length of the string

{

if (false ==(((char)Input[i]>='a'&& (char)Input[i]<='z') || ((char)Input[i] >= 'A' && ( char)Input[i] <= 'Z') || ((char)Input[i] == ' ') || ((char)Input[i] == '-') || ((char)Input[i] == '\''))) // if the char is a space

{

throw std::invalid\_argument("Non Valid Characters Detected"); // if not valid input throw invalid\_argument

}

}

return Input;

}

catch (std::invalid\_argument) // catch invalid argument say prompt and return the GetText Function

{

cout << "Invalid Symbol used. Ony letters \" \" and \"-\" are permitted. " << endl;

cout << "Please Re-enter the name : " << endl;

return GetText();

}

}

int main()

{

string address = "Data.csv"; // declare a source file

ofstream File; // start and output stream called file

string Name; // Declare the name Variable

int noOfPeople = 5; // declare an integer for number of people

for (int Person = 1; Person <=noOfPeople; Person++) // for number of people

{

cout << "Please enter name " << Person << ": " << endl;

Name = GetText();

File.open(address, ios::app); //get name input

Name = PrepName(Name); // Procees the name to the correct format

File << Name << endl; // Write the name to the file

cout << "Name Written to file" << endl;

File.close(); // close the file

}

cout << "The Name has been stored in CSV format where Collum 1 is the Given name, 2 the Middle Name(s) and 3 the Family name \n The File is stored as "<< address << endl;

//print a summary

return 0;

}