**TRƯỜNG ĐẠI HỌC SƯ PHẠM KỸ THUẬT TP. HỒ CHÍ MINH**



**KHOA ĐÀO TẠO CHẤT LƯỢNG CAO**

**NGÀNH CÔNG NGHỆ THÔNG TIN**

🞠◊🞠◊🞠



**BÁO CÁO ĐỒ ÁN**

**CẤU TRÚC DỮ LIỆU & GIẢI THUẬT**

**Đề tài: Xây dựng một chương trình từ điển**

**SVTH : NGUYỄN TRẦN TRUNG KIÊN**

**MSSV : 16110128**

**SVTH : VŨ TẤN ĐẠT**

**MSSV :**

**TP. Hồ Chí Minh, tháng 12 năm 2017**

MỤC LỤC

[NỘI DUNG 4](#_Toc500794100)

[1. Mô tả project 4](#_Toc500794101)

[1.1. Công dụng của chương trình 4](#_Toc500794102)

[1.2. Sơ đồ nội dung phần mềm 4](#_Toc500794103)

[1.3. Sơ đồ nội dung phần mềm 5](#_Toc500794104)

[2. Mô tả quá trình làm 5](#_Toc500794105)

[2.1. Thiết kế các Class 5](#_Toc500794106)

[2.2. Thiết kế giao diện 6](#_Toc500794107)

[a. Màn hình khóa 6](#_Toc500794108)

[b. Màn hình chính 7](#_Toc500794109)

[c. Form Add 8](#_Toc500794110)

[d. Form Delete 9](#_Toc500794111)

[e. Form Edit 10](#_Toc500794112)

[f. Form Option 10](#_Toc500794113)

[g. Form History 11](#_Toc500794114)

[h. Form Information 12](#_Toc500794115)

[2.3. Thiết kế các Class chính 13](#_Toc500794116)

[a. Class Node 13](#_Toc500794117)

[b. Class DSLK 14](#_Toc500794118)

[c. Class BangBam 17](#_Toc500794119)

[d. Class Word 20](#_Toc500794120)

[e. Class SpeakText 21](#_Toc500794121)

[2.4. Thiết kế các Code chính 22](#_Toc500794122)

[a. Hàm đọc File từ điển (input.txt) 22](#_Toc500794123)

[b. Hàm Load Data lên ComboBox 23](#_Toc500794124)

[c. Hàm Tra từ 24](#_Toc500794125)

[d. Hàm thêm từ 25](#_Toc500794126)

[e. Hàm xóa từ 26](#_Toc500794127)

[f. Hàm sửa từ 27](#_Toc500794128)

[3. Cái đặt và kiểm thử 28](#_Toc500794129)

[3.1. Cài đặt 29](#_Toc500794130)

[3.2. Thử nghiệm 29](#_Toc500794131)

[4. Phân công công việc 29](#_Toc500794132)

[5. Kết luận 30](#_Toc500794133)

[a. Kết quả 30](#_Toc500794134)

[b. Khó khăn 30](#_Toc500794135)

[c. Ý tưởng 31](#_Toc500794136)

[d. Ưu điểm 31](#_Toc500794137)

[e. Khuyết điểm 31](#_Toc500794138)

# **NỘI DUNG**

1. ***Mô tả project***
   1. **Công dụng của chương trình**

Trong thực tế, việc học tiếng anh vô cùng quan trọng đối với mỗi người. Đôi khi việc gặp những từ ngữ mới sẽ gặp nhiều khó khăn cho người dùng để tìm hiểu, vì vậy phần mềm này sẽ giúp cho việc tra cứu thuận tiện hơn.

* 1. **Sơ đồ nội dung phần mềm**
  2. **Sơ đồ nội dung phần mềm**

|  |  |
| --- | --- |
| **Tên Bảng** | **Chức năng** |
| Giao diện chính | \_Tra từ  \_Nghe từ  \_Liên kết các tính năng |
| Bảng Thêm | Thêm từ mới vào chương trình |
| Bảng Xóa | Xóa từ đã có trong chương trình |
| Bảng Sửa | Chỉnh sửa từ bị lỗi trong chương trình |
| Bảng Tùy Chọn | Lựa chọn xem một vài thông tin |
| Bảng Thông tin | Liệt kê tên đồ án, môn học và sinh viên thực hiện |
| Bảng Lịch sử | Xem lại những từ đã tra cứu |

1. ***Mô tả quá trình làm***
   1. **Thiết kế các Class**

|  |  |  |
| --- | --- | --- |
| **STT** | **Tên Class** | **Công dụng** |
| 1 | Word | Dùng để lưu trữ đối tượng chính của chương trình là từ, gồm có:  \_Name: Tên tiếng Anh của từ  \_Meaning: Giải nghĩa tiếng Việt từ  \_Explication: Giải thích nghĩa của từ |
| 2 | Node | Dùng để lưu trữ các đối tượng từ class Word |
| 3 | DSLK | Dùng để lưu trữ, liên kết các Node lại với nhau tạo thành một danh sách liên kết đơn |
| 4 | BangBam | Tập hợp nhiều danh sách liên kết có địa chỉ cụ thể từ việc băm nhằm lưu các Node để dễ dàng tra cứu từ |

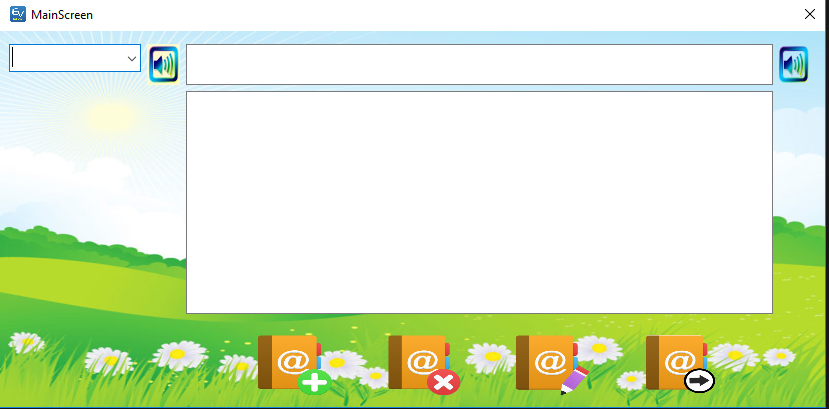
* 1. **Thiết kế giao diện**

1. **Màn hình khóa**

****

Yêu cầu người dùng bấm một phím bất kì không phải phím hệ thống để vô màn hình chính

1. **Màn hình chính**



Màn hình chính gồm:

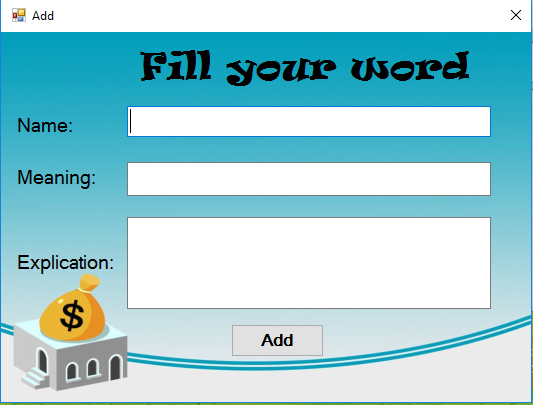
\_Một Combobox để lưu danh sách các từ

\_Hai Textbox để lưu nghĩa tiếng Việt của từ và Nghĩa của từ

\_Hai PictureBox hình cái Loa để kết nối mạng giúp cho việc nghe từ

\_Bốn PictureBox chức năng để thực hiện các thao tác trên từ điển

1. **Form Add**



Form Add dùng để thêm một từ mới gồm:

\_Bốn Label tiêu đề

\_Ba Textbox để chứa Tên từ, Nghĩa tiếng Việt và Giải nghĩa của từ

\_Một Button để kích hoạt chức năng thêm từ

1. **Form Delete**



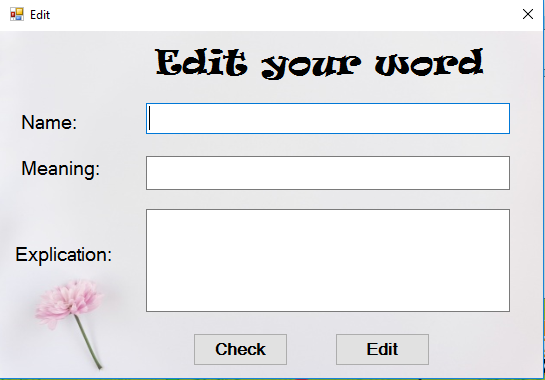
Form Delete dùng để xóa một từ bất kỳ trong từ điển, gồm có:

\_Hai Label tiêu đề

\_Một Textbox chứa Tên từ cần xóa

\_Một Button để kích hoạt chức năng xóa từ

1. **Form Edit**



Form Edit dùng để chỉnh sửa một từ bất kỳ, gồm có:

\_Bốn Label tiêu đề

\_Ba Textbox để chứa Tên từ, Nghĩa tiếng Việt và Giải nghĩa từ

\_Một Button để kiểm tra xem từ đó có trong từ điển hay chưa

\_Một Button để kích hoạt chức năng chỉnh sửa từ

1. **Form Option**



Form Option dùng để lựa chọn một vài yêu cầu

\_Xem lịch sử tra từ

\_Liên kết với một trang tra từ Online

\_Xem thông tin chương trình

Gồm:

\_Một Label giới thiệu

\_Ba PictureBox thể hiện ba chức năng

1. **Form History**



Form History dùng để lưu lại lịch sử tra từ, gồm:

\_Bốn Label giới thiệu

\_Một Combobox lưu các từ đã tra

\_Ba Textbox để lưu các thành phần của từ

1. **Form Information**



Form Information dùng để lưu một số thông tin của chương trình, gồm sáu Label giới thiệu

* 1. **Thiết kế các Class chính**

1. **Class Node**

public class Node

{

public Word word { get; set; }

public Node pNext { get; set; }

public Node()

{

word = new Word(null, null,null);

pNext = null;

}

public void CreateNode(Word wd)

{

word.Name = wd.Name;

word.Meaning = wd.Meaning;

word.Explication = wd.Explication;

pNext = null;

}

}

1. **Class DSLK**

public class DSLK

{

public Node Head;

public Node Tail;

public DSLK()

{

Head = Tail = null;

}

public void AddHead(Node p)

{

if (Head == null)

Head = Tail = p;

else

{

p.pNext = Head;

Head = p;

}

}

public void AddTail(Node p)

{

if (Head == null)

Head = Tail = p;

else

{

Tail.pNext = p;

Tail = p;

}

}

public Node Search(string s)

{

Node p = Head;

while (p != null)

{

if (p.word.Name == s)

return p;

p = p.pNext;

}

return null;

}

public Node SearchTruoc(string s)

{

Node p = Head.pNext;

Node q = Head;

if (Head != Tail)

{

while (p != null)

{

if (p.word.Name == s)

return q;

p = p.pNext;

q = q.pNext;

}

}

return null;

}

public void AddAfter(Node p, Node q)

{

try

{

if (Head == null)

throw new Exception("Can not Add");

q.pNext = p.pNext;

p.pNext = q;

if (p == Tail)

Tail = q;

}

catch (Exception e)

{

Console.WriteLine(e.Message);

}

}

public void RemoveHead()

{

try

{

if (Head == null)

throw new Exception("Empty List");

Node p = Head;

Head = p.pNext;

p = null;

}

catch (Exception e)

{

Console.WriteLine(e.Message);

}

}

public void RemoveAfter(Node p)

{

try

{

if (Head == null)

throw new Exception("Can not Remove");

Node q = p.pNext;

p.pNext = q.pNext;

if (p.pNext == null)

Tail = p;

}

catch (Exception e)

{

Console.WriteLine(e.Message);

}

}

}

1. **Class BangBam**

public class BangBam

{

public DSLK[] Table;

public static int DemDong()

{

FileStream file = new FileStream(

@"C:\Users\Trung Kien\Documents\GitHub\Do-An\DictionaryDemoFix\DictionaryDemo\input.txt",

FileMode.OpenOrCreate, FileAccess.ReadWrite);

StreamReader sr = new StreamReader(file);

string s = sr.ReadLine();

int d = 0;

do

{

d++;

s = sr.ReadLine();

}

while (s != null);

sr.Close();

file.Close();

return d;

}

int dong = DemDong();

public BangBam()

{

Table = new DSLK[dong];

for (int i = 0; i < dong; i++)

{

Table[i] = new DSLK();

}

}

public int Bam(string s)

{

int x = 1;

int n = s.Length;

for (int i = 0; i < n; i++)

{

if (s[i] < 'a' || s[i] > 'z')

i++;

else

x = x \* (s[i] - 96) % dong;

}

return x;

}

public void Add(Word wd)

{

int bam = Bam(wd.Name);

bool check = Search\_Check(wd.Name);

try

{

if (check == true)

throw new Exception("This word has been added");

Node p = new Node();

p.CreateNode(wd);

Table[bam].AddTail(p);

}

catch (Exception e)

{

Console.WriteLine(e.Message);

}

}

public bool Search\_Check(string s)

{

int bam = Bam(s);

Node p = new Node();

p = Table[bam].Head;

while (p != null)

{

if (p.word.Name == s)

return true;

p = p.pNext;

}

return false;

}

public Node Search\_Lookup(string s)

{

int bam = Bam(s);

bool check = Search\_Check(s);

try

{

if (check == false)

{

throw new Exception("HashTable doesn't have this word");

}

// sua

Node p = new Node();

p = Table[bam].Search(s);

return p;

}

catch (Exception e)

{

Console.WriteLine(e.Message);

return null;

}

}

public void RemoveWord(string s)

{

int bam = Bam(s);

if (s == Table[bam].Head.word.Name)

Table[bam].RemoveHead();

else

{

Node p = new Node();

p = Table[bam].SearchTruoc(s);

try

{

if (Table[bam].Head == null || p == null || Table[bam].Head == Table[bam].Tail || p.pNext == null)

throw new Exception();

Table[bam].RemoveAfter(p);

}

catch (Exception e)

{

Console.WriteLine(e);

}

}

}

public void EditWord(string name, string meaning, string explication)

{

int bam = Bam(name);

bool check = Search\_Check(name);

try

{

if (check == false)

{

throw new Exception("This word is not had in HashTable!!!");

}

Node p = new Node();

p = Table[bam].Head;

while (p != null)

{

if (p.word.Name == name)

{

p.word.Meaning = meaning;

p.word.Explication = explication;

}

p = p.pNext;

}

}

catch (Exception e)

{

Console.WriteLine(e.Message);

}

}

}

1. **Class Word**

public class Word:IComparable<Word>

{

private string name;

private string meaning;

private string explication;

public Word(string name, string meaning, string explication)

{

this.name = name;

this.meaning = meaning;

this.explication = explication;

}

public string Name

{

set { name = value; }

get { return name; }

}

public string Meaning

{

set { meaning = value; }

get { return meaning; }

}

public string Explication

{

set { explication = value; }

get { return explication; }

}

public int CompareTo(Word other)

{

return this.Name.CompareTo(other.Name);

}

}

1. **Class SpeakText**

public class SpeakText

{

private WebBrowser wbWeb;

public WebBrowser WbWeb

{

get { return wbWeb; }

set { wbWeb = value; }

}

public SpeakText(WebBrowser wb)

{

this.WbWeb = wb;

}

private void SetText(string data)

{

try

{

HtmlElement element = WbWeb.Document.GetElementById("text"); //Set thuộc tính của textbox trang web

element.SetAttribute("value", data);//Lấy giá trị của textbox trang web

}

catch (Exception e)

{

MessageBox.Show("Can not Access to Internet", "Error", MessageBoxButtons.OK, MessageBoxIcon.Error);

}

}

private void Speak()

{

try

{

HtmlElement element = WbWeb.Document.GetElementById("playbutton");//Set thuộc tính của button trang web

element.InvokeMember("click");//Khởi tạo sự kiện click của button

}

catch (Exception e)

{

}

}

public void Speaking(string data)

{

SetText(data);

Speak();

}

}

* 1. **Thiết kế các Code chính**

1. **Hàm đọc File từ điển (input.txt)**

public void LoadDataFile()

{

FileStream file = new FileStream(

@"C:\Users\Trung Kien\Documents\GitHub\Do-An\DictionaryDemoFix\DictionaryDemo\input.txt",

FileMode.OpenOrCreate, FileAccess.ReadWrite);

StreamReader sr = new StreamReader(file);

string s = sr.ReadLine();

do

{

string name = null, meaning = null, explication = null;

int j = 0;

while (j < s.Length && s[j] != '@')

{

name += s[j];

j++;

}

j++;

while (j < s.Length && s[j] != '@')

{

meaning += s[j];

j++;

}

j++;

while (j < s.Length)

{

explication += s[j];

j++;

}

Word wd = new Word(name, meaning, explication);

BB.Add(wd);

DS.Add(wd);

s = sr.ReadLine();

}

while (s != null);

sr.Close();

file.Close();

}

1. **Hàm Load Data lên ComboBox**

public void AddComboBox()

{

FileStream file = new FileStream(@"C:\Users\Trung Kien\Documents\GitHub\Do-An\DictionaryDemoFix\DictionaryDemo\input.txt",

FileMode.OpenOrCreate, FileAccess.ReadWrite);

StreamReader sr = new StreamReader(file);

string s = sr.ReadLine();

do

{

string name = null;

int j = 0;

while (j < s.Length && s[j] != '@')

{

name += s[j];

j++;

}

comboBox1.Items.Add(name);

s = sr.ReadLine();

}

while (s != null);

sr.Close();

file.Close();

}

1. **Hàm Tra từ**

public void comboBox1\_SelectedIndexChanged(object sender, EventArgs e)

{

ComboBox cb = sender as ComboBox;

string s = cb.Text;

Node k = BB.Search\_Lookup(s);

if (k != null)

{

textBox1.Text = k.word.Meaning;

textBox2.Text = k.word.Explication;

Word wd = new Word(s, textBox1.Text, textBox2.Text);

Node p = new Node();

p.CreateNode(wd);

LichSu.AddHead(p);

}

else

{

MessageBox.Show("Dictionay does not have this word", "Warning", MessageBoxButtons.OK,

MessageBoxIcon.Warning);

}

}

1. **Hàm thêm từ**

private void btn\_Add\_Click(object sender, EventArgs e)

{

string s1 = txt\_AddName.Text;

string s2 = txt\_AddMeaning.Text;

string s3 = txt\_AddExplication.Text;

try

{

if (s1 == "" || s2 == "")

throw new Exception("You have to fill enough word");

bool check = \_frmMainScreen.BB.Search\_Check(s1);

if (check == true)

MessageBox.Show("This word has been in this Dictionary", "Warning", MessageBoxButtons.OK,

MessageBoxIcon.Warning);

else

{

Word wd = new Word(s1, s2, s3);

\_frmMainScreen.BB.Add(wd);

\_frmMainScreen.comboBox1.Items.Clear();

\_frmMainScreen.DS.Add(wd);

\_frmMainScreen.DS.Sort();

MessageBox.Show("This word has been added", "Congratulation", MessageBoxButtons.OK,

MessageBoxIcon.Information);

FileStream fi = new FileStream(

@"C:\Users\Trung Kien\Documents\GitHub\Do-An\DictionaryDemoFix\DictionaryDemo\input.txt",

FileMode.Create, FileAccess.Write);

StreamWriter write = new StreamWriter(fi);

for (int i = 0; i < \_frmMainScreen.DS.Count; i++)

{

string k = \_frmMainScreen.DS[i].Name + '@' + \_frmMainScreen.DS[i].Meaning + '@' +

\_frmMainScreen.DS[i].Explication;

write.WriteLine(k);

}

write.Flush();

write.Close();

fi.Close();

\_frmMainScreen.AddComboBox();

\_frmMainScreen.Loading();

}

}

catch (Exception exception)

{

MessageBox.Show(exception.Message, "Error", MessageBoxButtons.OK, MessageBoxIcon.Error);

}

}

1. **Hàm xóa từ**

private void btn\_Del\_Click(object sender, EventArgs e)

{

string s1 = txt\_DelName.Text;

try

{

if (s1 == "" || s1 == null)

throw new Exception("You have to fill enough word");

bool check = \_frmMainScreen.BB.Search\_Check(s1);

if (check == false)

MessageBox.Show("This word is not in this Dictionary", "Warning", MessageBoxButtons.OK,

MessageBoxIcon.Warning);

else

{

\_frmMainScreen.BB.RemoveWord(s1);

MessageBox.Show("This Word has been Delete", "Congratulation", MessageBoxButtons.OK,

MessageBoxIcon.Information);

\_frmMainScreen.comboBox1.Items.Remove(s1);

for (int i = 0; i < \_frmMainScreen.DS.Count; i++)

{

if (\_frmMainScreen.DS[i].Name == s1)

\_frmMainScreen.DS.RemoveAt(i);

}

FileStream file = new FileStream(

@"C:\Users\Trung Kien\Documents\GitHub\Do-An\DictionaryDemoFix\DictionaryDemo\input.txt",

FileMode.Create, FileAccess.Write);

StreamWriter write = new StreamWriter(file);

for (int i = 0; i < \_frmMainScreen.DS.Count; i++)

{

string k = \_frmMainScreen.DS[i].Name + '@' + \_frmMainScreen.DS[i].Meaning + '@' +

\_frmMainScreen.DS[i].Explication;

write.WriteLine(k);

}

write.Flush();

write.Close();

file.Close();

}

}

catch (Exception exception)

{

MessageBox.Show(exception.Message, "Error", MessageBoxButtons.OK, MessageBoxIcon.Error);

}

}

1. **Hàm sửa từ**

private void btn\_Edit\_Click(object sender, EventArgs e)

{

try

{

if (txt\_EditName.Text == "")

throw new Exception("You have to fill an right word");

\_frmMainScreen.BB.EditWord(txt\_EditName.Text, txt\_EditMeaning.Text, txt\_EditExplication.Text);

MessageBox.Show("This word has been edited", "Congratulation", MessageBoxButtons.OK, MessageBoxIcon.Information);

for (int i = 0; i < \_frmMainScreen.DS.Count; i++)

{

if (\_frmMainScreen.DS[i].Name == txt\_EditName.Text)

{

\_frmMainScreen.DS[i].Name = txt\_EditName.Text;

\_frmMainScreen.DS[i].Meaning = txt\_EditMeaning.Text;

\_frmMainScreen.DS[i].Explication = txt\_EditExplication.Text;

}

}

FileStream file = new FileStream(

@"C:\Users\Trung Kien\Documents\GitHub\Do-An\DictionaryDemoFix\DictionaryDemo\input.txt",

FileMode.Create, FileAccess.Write);

StreamWriter write = new StreamWriter(file);

for (int i = 0; i < \_frmMainScreen.DS.Count; i++)

{

string k = \_frmMainScreen.DS[i].Name + '@' + \_frmMainScreen.DS[i].Meaning + '@' +

\_frmMainScreen.DS[i].Explication;

write.WriteLine(k);

}

write.Flush();

write.Close();

file.Close();

}

catch (Exception exception)

{

MessageBox.Show(exception.Message, "Error", MessageBoxButtons.OK, MessageBoxIcon.Error);

}

}

1. ***Cái đặt và kiểm thử***
   1. **Cài đặt**

-Chương trình được cái đặt trên nền Visual Studio 2017, sử dụng ngôn ngữ C# và Project Window Form để thực hiện

-Có sử dụng file input.txt để chứa dữ liệu, thực hiện đọc file trong Project

-Hàm băm được thực hiện theo công thức: lấy các kí tự trong chuỗi trừ đi 96 rồi nhân lại với nhau

-File txt có bao nhiêu từ thì Bảng băm sẽ có bấy nhiêu danh sách liên kết đơn để chứa từ (dung lượng bộ nhớ có thề cho phép lượng từ hơn 10000 từ)

-Sử dụng 2 link để đọc từ

+ <http://responsivevoice.org/text-to-speech-languages/van-ban-de-noi-trong-tieng-viet/>

+ <http://responsivevoice.org/text-to-speech-languages/us-english-text-to-speech/>

-Tốc độ tìm kiếm là O(1), tốn gấp n lần bộ nhớ (n là số từ)

* 1. **Thử nghiệm**

- Chọn một từ bên ComboBox thì hai Textbox sẽ hiện lên phần tiếng Việt và giải thích nghĩa

- Khi ComboBox có từ thì hai loa sẽ phát âm

- Khi không có mạng thì bấm vào loa sẽ thông báo

- Cách băm :

Ví dụ: computer

gọi x = 1 là vị trí trong bảng băm

b1: x = x \* 3(vị trí c trong bảng chữ cái) % 85(số từ có trong file input) = 3

b2: x = x \* 15(vị trí o trong bảng chữ cái) % 85(số từ có trong file input) = 45

b3: x = x \* 13(vị trí m trong bảng chữ cái) % 85(số từ có trong file input) = 75

b4: x = x \* 16(vị trí p trong bảng chữ cái) % 85(số từ có trong file input) = 10

b5: x = x \* 21(vị trí u trong bảng chữ cái) % 85(số từ có trong file input) = 40

b6: x = x \* 20(vị trí t trong bảng chữ cái) % 85(số từ có trong file input) = 35

b7: x = x \* 5(vị trí e trong bảng chữ cái) % 85(số từ có trong file input) = 5

b8: x = x \* 18(vị trí r trong bảng chữ cái) % 85(số từ có trong file input) = 5

kết quả: băm được tại vị trí 5

- Bảng băm sau khi thực hiện trên file input.txt:

0: queen->x-ray 1:

2: 3:

4: spring 5: computer->desktop->water

6: 7:

8: 9:

10: out->ride->sea 11:

12: 13:

14: bag 15: anxious->day->divide->tea->under

16: add 17:

18: gun 19:

20: book->get->market->nice->red->word 21: animal

22: run 23:

24: 25: ghost->what

26: 27:

28: ban 29:

30: football 31:

32 : 33:

34: 35: destroy->earth->option->windy

36: 37:

38: 39: kiss

40: bed->hello->sky->star->sword->year 41: big->fill

42: bill 43:

44: 45: parent->visible

46: 47:

48: fish 49:

50: bee->beer->rice->super market->volcano 51:

52: 53:

54: 55: bike->clock->river

56: 57:

58: rain 59: milk

60: apple->cat->eyes->lies->name->power 61: kill

62: salad 63: chair

64: 65: forest->table tennis

66: 67: ->club

68: 69:

70: mom->telephone->zoo 71:

72: 73:

74: 75: dictionary->egg->lock

76: sad 77: miss

78: 79:

80: count->hide->language->love->umbrella 81:

82: 83:

84: fan->stadium

1. ***Phân công công việc***

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **STT** | **Mô tả công việc** | **Nguyễn Trần Trung Kiên** | **Vũ Tấn Đạt** | **Chi tiết** |
| 1 | Thiết kế Class Word |  | x |  |
| 2 | Thiết kế Class DSLK | x | x | 50-50 |
| 3 | Thiết kế Class BangBam | x | x | 50-50 |
| 4 | Thực hiện phần đọc file input.txt |  | x |  |
| 5 | Thiết kế Form LockScreen | x |  |  |
| 6 | Thiết kế Form MainScreen | x |  |  |
| 7 | Thiết kế các Form Add, Delete, Edit, Option |  | x |  |
| 8 | Thiết kế Form History và Form Information |  | x |  |
| 9 | Thiết kế phần nghe từ | x |  |  |
| 10 | Làm phần hình ảnh cho các Form | x |  |  |
| 11 | Thực hiện hàm thêm, xóa, sửa | x | x | 50-50 |

1. ***Kết luận***
2. **Kết quả**

Sau khi trải qua nửa học kì nghiên cứu cũng như thực hiện với sự giúp đỡ của thầy Trần Công Tú, nhóm chúng em đã hoàn thành được một chương trình từ điển Anh-Việt ứng dụng kiến thức môn Cầu trúc dữ liệu và Giải thuật là Bảng băm

1. **Khó khăn**

-Do chúng em lấy điểm phần bài tập khá nhiều và lịch học các môn dày đặc nên chưa tập trung hơn 100% sức lực vào phần đồ án này.

-Do tụi em chưa được làm quen với Github nên hai đứa thường gửi bài cho nhau qua lại thay phiên nhau làm, và có vấn đề trục trặc về Github, chúng em không biết lỗi gì nên file Github hình như bị sự cố, nếu không có Link mong thầy thông cảm.

-Có một vài sự sai sót khi thiết kế hàm Băm nhưng nhờ thầy Trần Công Tú mà tụi em đã khắc phục và tìm ra cách giải quyết tốt hơn.

1. **Ý tưởng**

-Nếu có nhiều thời gian nghiên cứu hơn thì chúng em có thề dùng cơ sở dữ liệu (database) để lưu danh sách từ thì code sẽ ngắn gọn hơn và thao tác cũng sẽ dễ dàng hơn.

-Có thế phát triển từ điển này theo công nghệ mới của .NET có thế lấy được giọng nói thì ta có thể liên kết âm thanh với chương trình để dễ dàng sử dụng thông qua lời nói, không cần phải gõ phím hoặc bấm chuột.

1. **Ưu điểm**

Phần mềm có giao diện khá đẹp, gần gũi với người dùng, tốc độ nhanh thuận tiện cho việc tra cứu, nếu thiếu sót người dùng có thề thêm, xóa hay sửa từ cần thiết và đặc biệt có thể nghe cách phát âm của từ đó.

1. **Khuyết điểm**

Khi đọc từ từ file thì dữ liệu load lên sẽ không nhiều và nhanh được như database và không mở rộng được nhiều vốn từ chuyên ngành sẽ khó cho những người dùng có trình độ và vốn từ cao, chỉ thích hợp cho những người dùng bình thường.