

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

XÂY DỰNG TỪ ĐIỂN SỬ DỤNG BẢNG BĂM

SVTH : Đỗ Quốc Hùng
MSSV : 16110097
SVTH : Huỳnh Trần Thái Bảo
MSSV : 16110015

TP. Hồ Chí Minh, tháng 12 năm 2019

MỤC LỤC

LỜI CẢM ƠN	1
I. PHẦN MỞ ĐẦU	2
1. Lí do và mục đích chọn đề tài	2
2. Mục tiêu cho đề tài	2
3. Giới thiệu đề tài Từ Điển	3
II. PHẦN NỘI DUNG	4
1. Cơ sở lý thuyết	4
1.1. Thế nào là bảng băm ?	4
1.2. Làm sao để chọn một hàm băm tốt ?	4
2. Mô tả ứng dụng	5
2.1. Chức năng phần mềm.	5
2.2. Các chức năng thêm	5
3. Quá trình thực viết chương trình	5
3.1. Giao diện của cửa form chính	5
3.2. Code của giải thuật bảng băm	6
III. KẾT LUẬN	14
1. Bảng phân công kế hoạch	14
2. Đánh giá ưu và nhược điểm của quá trình làm	15
2.1. Kết quả đạt được	15
2.2. Ưu điểm	15
2.3. Nhược điểm	15
IV. TÀI LIỆU THAM KHẢO	16

DANH MỤC HÌNH

Hình 1 Hình form giao diện chính	5
Hình 2 Hình form giao diện thêm từ	6

DANH MỤC BẢNG

Bảng 1: Code class Dictionary	8
Bảng 2: Code class Word.....	10
Bảng 3: Code class Dataflow	13
Bảng 4: Bảng phân công công việc	14

LỜI CẢM ƠN

Trong thời gian làm đồ án môn cấu trúc dữ liệu, em đã nhận được nhiều sự giúp đỡ, đóng góp ý kiến và chỉ bảo nhiệt tình của thầy và bạn bè. Em xin gửi lời cảm ơn chân thành đến Ths. Trần Công Tú, giảng viên phụ trách Bộ môn Cấu trúc dữ liệu và giải thuật người đã tận tình hướng dẫn, chỉ bảo em trong suốt quá trình làm đồ án.

Để hoàn thành đồ án “TỪ ĐIỂN SỬ DỤNG BẢNG BĂM”, nhóm em đã khảo sát, thu nhập dữ liệu và tổng hợp thông tin từ nhiều hướng khác nhau. Nhưng quá trình xây dựng chương trình, nghiên cứu và cố gắng hoàn thành, vì trình độ còn hạn chế nên có nhiều khó khăn làm chậm tiến độ. Mặt khác, đề tài tuy đã có sự giúp đỡ của GVHD nhưng vẫn không thể tránh khỏi những sai sót.

Do đó, nhóm em mong sẽ nhận được sự góp ý chân thành của thầy và các bạn đọc. Những ý kiến đóng góp về đề tài sẽ được nhóm em tiếp nhận để có thêm kinh nghiệm khi làm các đề tài nghiên cứu khác. Một lần nữa, nhóm em xin cảm ơn Thầy Trần Công Tú và các bạn.

I. PHẦN MỞ ĐẦU

1. Lí do và mục đích chọn đề tài

Dựa vào những kiến thức đã học từ những môn nhập môn lập trình và kỹ thuật lập trình, em muốn làm ra một chương trình từ điển đơn giản kết hợp với window form,...

Mục đích chính khi nhóm em chọn đề tài này vì: Muốn tạo một ứng dụng bổ ích có áp dụng kiến thức đã học đồng thời sau đó có thể hỗ trợ việc học ngoại ngữ.

Với chức năng phát âm, chúng ta có thể nghe được người bản xứ phát âm từ mà chúng ta muốn, đồng thời chúng ta có thể thêm từ mới vào kho từ vựng để hoàn thiện từ điển của riêng mình. Ngoài ra có thể phát triển thêm đa ngôn ngữ.

Từ những đặc điểm trên nên nhóm em quyết định xây dựng ứng dụng Từ Điển Anh - Việt sử dụng bảng băm.

2. Mục tiêu cho đề tài

Áp dụng được bảng băm vào từ điển khi tra cứu từ vựng với các chức năng :

- Tìm kiếm từ vựng
- Phát âm
- Thêm từ
- Xóa từ

3. Giới thiệu đề tài Từ Điển

Từ điển là một công cụ hỗ trợ giao tiếp đã quen thuộc với mọi người khi học ngoại ngữ. Đa số chúng ta sẽ nghĩ đó là một cuốn sách dày cộm. Nhưng trong thế giới công nghệ 4.0 hiện nay, thì mọi thứ được lập trình sẵn trên máy tính hoặc điện thoại thông minh phổ biến và lớn mạnh. Và từ điển cũng được chúng ta xây dựng trên công nghệ thông qua ngôn ngữ lập trình winform C#.

II. PHẦN NỘI DUNG

1. Cơ sở lý thuyết

1.1. Thế nào là bảng băm ?

Trong khoa học máy tính, bảng băm là một cấu trúc dữ liệu sử dụng hàm băm để ánh xạ từ giá trị xác định, được gọi là khóa (ví dụ như tên của một người), đến giá trị tương ứng (ví dụ như số điện thoại của họ). Do đó, bảng băm là một mảng kết hợp. Hàm băm được sử dụng để chuyển đổi từ khóa thành chỉ số (giá trị băm) trong mảng lưu trữ các giá trị tìm kiếm.

1.2. Làm sao để chọn một hàm băm tốt ?

Hàm băm tốt và thuật toán tốt là hai yếu tố cần thiết để bảng băm hoạt động hiệu quả, nhưng không dễ để đạt được.

Một yêu cầu cơ bản là hàm băm nên phân bố đều các giá trị băm trong mảng. Phân bố không đều làm tăng số lượng va chạm, và do đó tăng chi phí giải quyết chúng. Tính đồng đều đôi khi khó có thể đảm bảo trong thiết kế, nhưng có thể được đánh giá trong thực tiễn bằng cách sử dụng các bài kiểm tra thống kê.

Trong phương pháp địa chỉ mở, các hàm băm cũng nên tránh hiện tượng phân nhóm (ánh xạ hai hay nhiều khóa đến các vị trí liên tiếp trong mảng). Phân nhóm như vậy có thể khiến chi phí tra cứu tăng vọt, ngay cả khi hệ số đầy thấp và va chạm là không thường xuyên.

Các hàm băm mật mã học được cho là hàm băm tốt cho bất kỳ kích thước bảng nào, hoặc bằng cách tính số dư hoặc bằng mặt nạ bit. Tuy nhiên, những phẩm chất này khó có thể bù lại chi phí tính toán lớn hơn nhiều và sự phức tạp của thuật toán.

2. Mô tả ứng dụng

2.1. Chức năng phần mềm.

Phần mềm có các chức năng cơ bản của một từ điển:

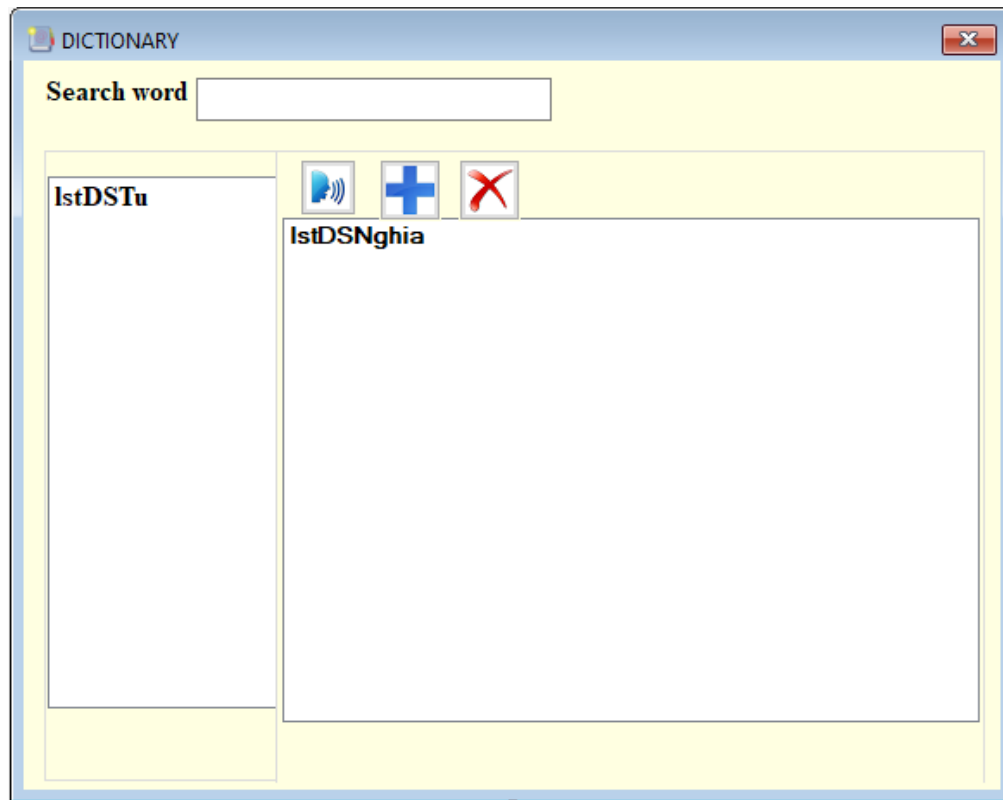
- Search từ vựng: hiện ra cách phát âm, nghĩa của từ.
- Phát âm: Nghe được giọng của người bản xứ phát âm từ.
- Thêm, xóa từ: có thể thêm hoặc xóa từ vựng của từ điển.

2.2. Các chức năng thêm

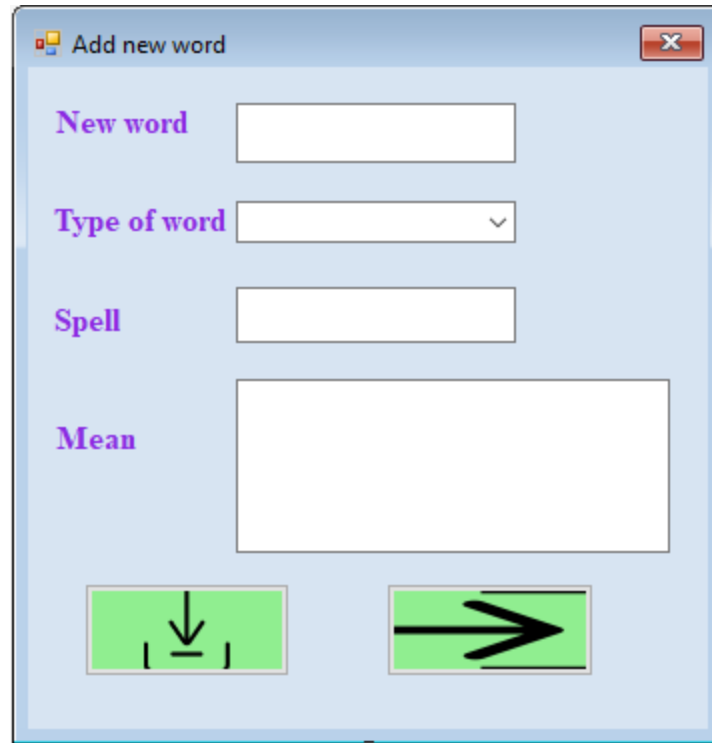
Ngoài chức năng trên thì còn các chức năng đang hoàn thiện dần như: sao chép, dán, xem các từ tra gần đây, từ điển Việt - Anh,...

3. Quá trình thực viết chương trình

3.1. Giao diện của cửa form chính



Hình 1 Hình form giao diện chính

A screenshot of a Windows-style dialog box titled "Add new word". The dialog has a light blue background and a standard title bar with a close button (X). Inside, there are four input fields arranged vertically. The first is labeled "New word" in purple. The second is labeled "Type of word" in purple and is a dropdown menu. The third is labeled "Spell" in purple. The fourth is labeled "Mean" in purple and is a larger text area. At the bottom of the dialog, there are two green buttons. The left button has a black downward arrow icon, and the right button has a black rightward arrow icon.

Hình 2 Hình form giao diện thêm từ

3.2.Code của giải thuật bảng băm

Nhóm thực hiện xây dựng bảng băm dựa trên bảng chữ cái tiếng anh có 26 ký tự, theo bảng mã ASCII ứng với giá trị từ 97 đến 122. Vì vậy sử dụng hàm hashkey (string key) lấy mã ASCII ký tự đầu của word chia lấy phần dư cho 97. Ta sẽ được một mảng gồm các số từ 0 đến 25. Thay vì thông thường ta dùng danh sách liên kết để lưu thông tin của word, ở đây lưu tất cả xuống file *.txt để thao tác quản lý. Ứng với 26 ký tự từ a đến z cũng như giá trị từ 0 đến 25 là 26 file. Mỗi file sẽ lưu các word có ký tự đầu ứng với mã hash.

```

class Dictionary
{
    // Khai báo 1 mảng dataflow gồm 26 ô ứng với 26 ký tự trong bảng alphabet
    public Dataflow[] ds = new Dataflow[26];
    // Khai báo 1 mảng chứa tên file chứa dữ liệu ứng với từng ký tự
    private string[] dsFileName = new string[26];

    //Khởi tạo bộ từ điển từ tất cả các file hiện lên trên list từ
    public Dictionary()
    {
        //Gán giá trị là tên file ứng với thứ tự trong mảng
        dsFileName[0] = "a.txt"; dsFileName[1] = "b.txt"; dsFileName[2] = "c.txt";
dsFileName[3] = "d.txt";
        dsFileName[4] = "e.txt"; dsFileName[5] = "f.txt"; dsFileName[6] = "g.txt";
dsFileName[7] = "h.txt";
        dsFileName[8] = "i.txt"; dsFileName[9] = "j.txt"; dsFileName[10] = "k.txt";
dsFileName[11] = "l.txt";
        dsFileName[12] = "m.txt"; dsFileName[13] = "n.txt"; dsFileName[14] =
"o.txt"; dsFileName[15] = "p.txt";
        dsFileName[16] = "q.txt"; dsFileName[17] = "r.txt"; dsFileName[18] =
"s.txt"; dsFileName[19] = "t.txt";
        dsFileName[20] = "u.txt"; dsFileName[21] = "v.txt"; dsFileName[22] =
"w.txt"; dsFileName[23] = "x.txt";
        dsFileName[24] = "y.txt"; dsFileName[25] = "z.txt";
        for (int i = 0; i < 26; i++)
        {
            ds[i] = new Dataflow(dsFileName[i]);
        }
    }
    // Khởi tạo một dataflow từ một số hash được.
    public Dataflow this[int index]
    {
        get { return (Dataflow)ds[index]; }
    }
    // Thao tác dữ liệu trên file
    // Truyền dữ liệu vào file
    public void WriteToData()
    {
        foreach (Dataflow a in ds)
            a.WriteFile();
    }
    //Xóa dữ liệu
    public void xoadata()
    {
        foreach (Dataflow a in ds)
            a.xoatrang();
    }
    //Đọc dữ liệu
    public void ReadFromData()
    {
        foreach (Dataflow a in ds)
            a.ReadFile();
    }
    // Hàm hash: Do bảng chữ cái có 26 ký tự, bắt đầu từ các ký tự a....z, theo
    bảng ascii ứng với 97...122
    public int hashkey(string key)
    {
        return ((int)key[0])%97;
    }
}

```

```

    }
    //Lấy kết quả từ mã hash trong ds
    public string writedata(string key)
    {
        string kq = "";
        int i = hashkey(key);
        Console.WriteLine("Char " + key + " hash to: " + i);
        //Ứng với danh sách mã hash rồi search từ đó bằng key
        kq = ds[i].SearchWord(key);
        return kq;
    }
    //Hàm search filter, truyền một chuỗi string và trả về một mảng chứa từ khóa đó
    public void HienThiTuLienQuan(string key, ref string[] mangchuakey)
    {
        int i = hashkey(key);
        Console.WriteLine("Char " + key + " hash to: " + i);
        string[] keys = new string[50];
        int dem;
        ds[i].hienthitucolienquan(key, ref keys, out dem);
        for (int j = 0; j < dem; j++)
        {
            mangchuakey[j] = keys[j];
        }
    }
    //Xóa từ
    public bool delete(string key)
    {
        //Hash lấy số thứ tự của ds
        int i = hashkey(key);
        //Nếu tìm thấy từ thì sẽ xóa
        if (ds[i].timkiem(key) == true)
        {
            ds[i].xoa(key);

            return true;
        }
        //Nếu không tìm thấy trả về null
        else return false;
    }
    //Thêm từ vào danh sách
    public bool themtu(string key, string tuloai, string phatam, string nghĩa)
    {
        //Lấy mã hash
        int i = hashkey(key);
        Console.WriteLine("Char " + key + " hash to: " + i);
        if (ds[i].timkiem(key) == false)
        {
            ds[i].AddWord(key, tuloai, phatam, nghĩa);
            return true;
        }
        else return false;
    }
}

```

Bảng 1: Code class Dictionary

```

class Word
{
    private string key;
    //get set
    public string Key
    {
        get { return key; }
        set { key = value; }
    }
    //Mỗi từ bao gồm ba thành phần: từ loại, phát âm và nghĩa
    public string tuloi;
    public string phatam;
    public string nghĩa;

    //Khởi tạo từ loại
    public string Tuloi
    {
        get { return tuloi; }
        set { tuloi = value; }
    }
    //Khởi tạo phát âm
    public string Phatam
    {
        get { return phatam; }
        set { phatam = value; }
    }
    //Khởi tạo nghĩa
    public string Nghia
    {
        get { return nghĩa; }
        set { nghĩa = value; }
    }
    //Hàm khởi tạo mặc định
    public Word()
    {
    }
    //Hàm khởi tạo một từ mới, key chính là mã hash
    public Word(string k, string tl, string pa, string ngh)
    {
        key = k;
        tuloi = tl;
        phatam = pa;
        nghĩa = ngh;
    }
    //Hàm khởi tạo với key string
    public Word(string k)
    {
        key = k;
    }
}

```

```

//int ra
//Hàm nối chuỗi xuất kết quả đầy đủ của một từ
public override string ToString()
{
    string s = "";
    s = tuloai + phatam + nghĩa;
    return s;
}
}

```

Bảng 2: Code class Word

```

class Dataflow
{
    //Khởi tạo danh sách hashtable
    public Hashtable ds = new Hashtable();
    //Tên của file
    public string fileName = "";

    //Lớp khởi tạo mặc định
    public Dataflow()
    {
    }
    //Lớp khởi tạo truyền vào tên file
    public Dataflow(string tenfile)
    {
        fileName = tenfile;
    }
    //Lấy một từ từ mã hash trong ds
    public Word this[int cs]
    {
        get { return (Word)ds[cs]; }
    }
    //Thêm một từ bao gồm khóa và từ
    public void AddWord(string key, Word data)
    {
        ds.Add(key, data);
    }
    //Thêm một từ bao gồm đầy đủ thông tin
    public void AddWord(string key, string tuloai, string phatam, string nghĩa)
    {
        Word a = new Word(key, tuloai, phatam, nghĩa);
        ds.Add(key, a);
    }
    //Thao tác với file txt
    //Hàm write file
    public void WriteFile()
    {
        //Nếu file tồn tại thì xóa file đó.
        if (File.Exists(fileName)) File.Delete(fileName);
        //Tạo một file mới
        FileStream myFile = new FileStream(fileName, FileMode.Create);
        //Viết lại file
        StreamWriter sw = new StreamWriter(myFile);
        //Khởi tạo một từ để chuẩn bị viết vào
        Word w = new Word();
    }
}

```



```

        string s = w.tuloai + w.phatam + w.nghia;

        foreach (DictionaryEntry a in ds)
        {
            //Ghi xuống file
            sw.WriteLine(a.Key + ":" + a.Value + ":" + w.tuloai + a.Value + ":" +
w.phatam + a.Value + w.nghia);
            //Đẩy các dữ liệu còn lại xuống file
            sw.Flush();
        }
        //Đóng kết nối streamwriter
        sw.Close();
        //Đóng file
        myFile.Close();
    }

    //Xóa dữ liệu trong 1 trang
    //Xóa bằng cách truyền chuỗi rỗng vào file mới
    public void xoatrang()
    {

        //Nếu file tồn tại thì xóa file đó.
        if (File.Exists(fileName)) File.Delete(fileName);
        //Tạo một file mới
        FileStream myFile = new FileStream(fileName, FileMode.Create);
        //Viết lại file
        StreamWriter sw = new StreamWriter(myFile);
        //Khởi tạo một từ để chuẩn bị viết vào
        Word w = new Word();
        //Chuẩn bị một chuỗi rỗng
        string s = "";

        foreach (DictionaryEntry a in ds)
        {
            sw.WriteLine(s);
            sw.Flush();
        }

        sw.Close();
        myFile.Close();
    }
    //Đọc file
    public void ReadFile()
    {
        string text;
        string[] tam;
        FileStream myFile;
        if (File.Exists(fileName))
        {
            myFile = new FileStream(fileName, FileMode.Open);
            StreamReader sr = new StreamReader(myFile);
            for (; (text = sr.ReadLine()) != null;)
            {
                tam = text.Split(':');
                AddWord(tam[0], tam[1], tam[2], tam[3]);
            }
            sr.Close();
            myFile.Close();
        }
    }

```

```

    }
    else myFile = new FileStream(fileName, FileMode.Create);
}
//Tìm kiếm từ trong file bằng một chuỗi string
public string SearchWord(string key)
{
    string s = "";
    ICollection c = ds.Keys; //Danh sách từ
    foreach (string item in c) //Vòng lặp qua từng từ
    {
        //Tìm các từ giống với chuỗi nhập vào
        if (item == key)
        {
            s = item + " : " + "\n" + ds[item].ToString();
            break;
        }
    }
    //Trả về chuỗi tìm được
    return s;
}
//Hàm in kết quả ra màn hình
public override string ToString()
{
    string kq = "";
    //Tạo một từ điển enumerator để lấy dữ liệu từ danh sách
    IDictionaryEnumerator enumerator = ds.GetEnumerator();
    while (enumerator.MoveNext()) // xuất tất cả kết quả tìm được
    {
        kq = enumerator.Key + " : " + enumerator.Value.ToString();
    }
    return kq;
}
// Hàm xóa
public Boolean xoa(string key)
{
    if (ds.ContainsKey(key))
    {
        ds.Remove(key);
        return true;
    }
    else
        return false;
}

//Hàm tìm kiếm
public Boolean timkiem(string key)
{
    if (ds.ContainsKey(key))
        return true;
    else return false;
}
//
public string intoanbo()
{
    string s = "";
    ICollection c = ds.Keys;
    foreach (string item in c)

```

```

        {
            s = s + item + "\n";
        }
        return s;
    }
    //
    public void inkey(ref string[] s)
    {
        int i = 0;
        s = new string[ds.Keys.Count];
        ICollection c = ds.Keys;
        foreach (string item in c)
        {
            s[i] = item;
            ++i;
        }
    }
    //Hàm search filter
    public void hienthitucolienquan(string key, ref string[] keys, out int dem)
    {
        dem = 0;
        int i = 0;
        ICollection c = ds.Keys;
        foreach (string item in c)
        {
            if (item.Contains(key))
            {
                keys[i] = item;
                i++;
                dem++;
            }
        }
    }
}

```

Bảng 3: Code class Dataflow

III. KẾT LUẬN

1. Bảng phân công kế hoạch

NỘI DUNG CHI TIẾT CÔNG VIỆC	Huỳnh Trần Thái Bảo	Đỗ Quốc Hùng	Ngày bắt đầu (dự kiến)	Ngày kết thúc (dự kiến)	Ngày bắt đầu (thực tế)	Ngày kết thúc (thực tế)
Tìm hiểu winform	X	X	5-9-2019	10-9-2019	5-9-2019	10-9-2019
Lên thiết kế giao diện		X	11-9-2019	20/13/2019	11-9-2019	20/13/2019
Triển khai	X	X	21-9-2019	22-9-2019		
Tìm hiểu mảng băm	X	X	21-9-2019	23-9-2019	21-9-2019	22/9/2019
Xây dựng từ điền	X	X	24/9/2019	30/9/2019	22/9/2019	30/9/2019
Xây dựng chức năng tìm kiếm từ	X		24/9/2019	30/9/2019	22/9/2019	15-10- 2019
Xây dựng chức năng thêm từ	X	X	25/9/2019	30/9/2019	25/9/2019	14-10- 2019
gặp giáo viên để xin ý kiến	X	X	4-10-2019	6-10-2019	8-10-2019	23-10- 2019
chỉnh sửa code mạch lạc rõ ràng và những lỗi nhỏ	X	X	4-10-2019	6-10-2019	8-10-2019	23-10- 2019

Bảng 4: Bảng phân công công việc

2. Đánh giá ưu và nhược điểm của quá trình làm

2.1. Kết quả đạt được

Hoàn thành các chức năng cơ bản của 1 từ điển như: search từ vựng, phát âm, thêm từ, xóa từ.

Ứng dụng được bảo đảm trong việc thao tác dữ liệu.

2.2. Ưu điểm

- Với phần mềm này người sử dụng có thể tra cứu nghĩa của từ vì giao diện rất dễ sử dụng. Mặt khác có thể thêm vào những từ, cụm từ mới vào kho từ vựng của mình một cách nhanh chóng.
- Chế độ phát âm có thể cho người sử dụng luyện tập được cách đọc đúng từ vựng cho việc giao tiếp.

2.3. Nhược điểm

- Chương trình chưa tối ưu được hết các tính năng của của từ điển như chỉnh sửa từ, xóa từ.
- Phần giao diện thiết kế chưa được đẹp.
- Chương trình chưa có chức năng dịch nghĩa ngược lại từ tiếng Việt sang tiếng Anh.

IV. TÀI LIỆU THAM KHẢO

<https://www.howkteam.vn/course/tao-chuc-nang-doc-chu-tu-dien-noi-c-winform/tao-cau-truc-luu-tru-du-lieu-tu-dien-noi-c-winform-130> - tham khảo cấu trúc của một từ điển sử dụng bảng băm trên c#.

https://vi.wikipedia.org/wiki/B%E1%BA%A3ng_b%C4%83m – lý thuyết về bảng băm.