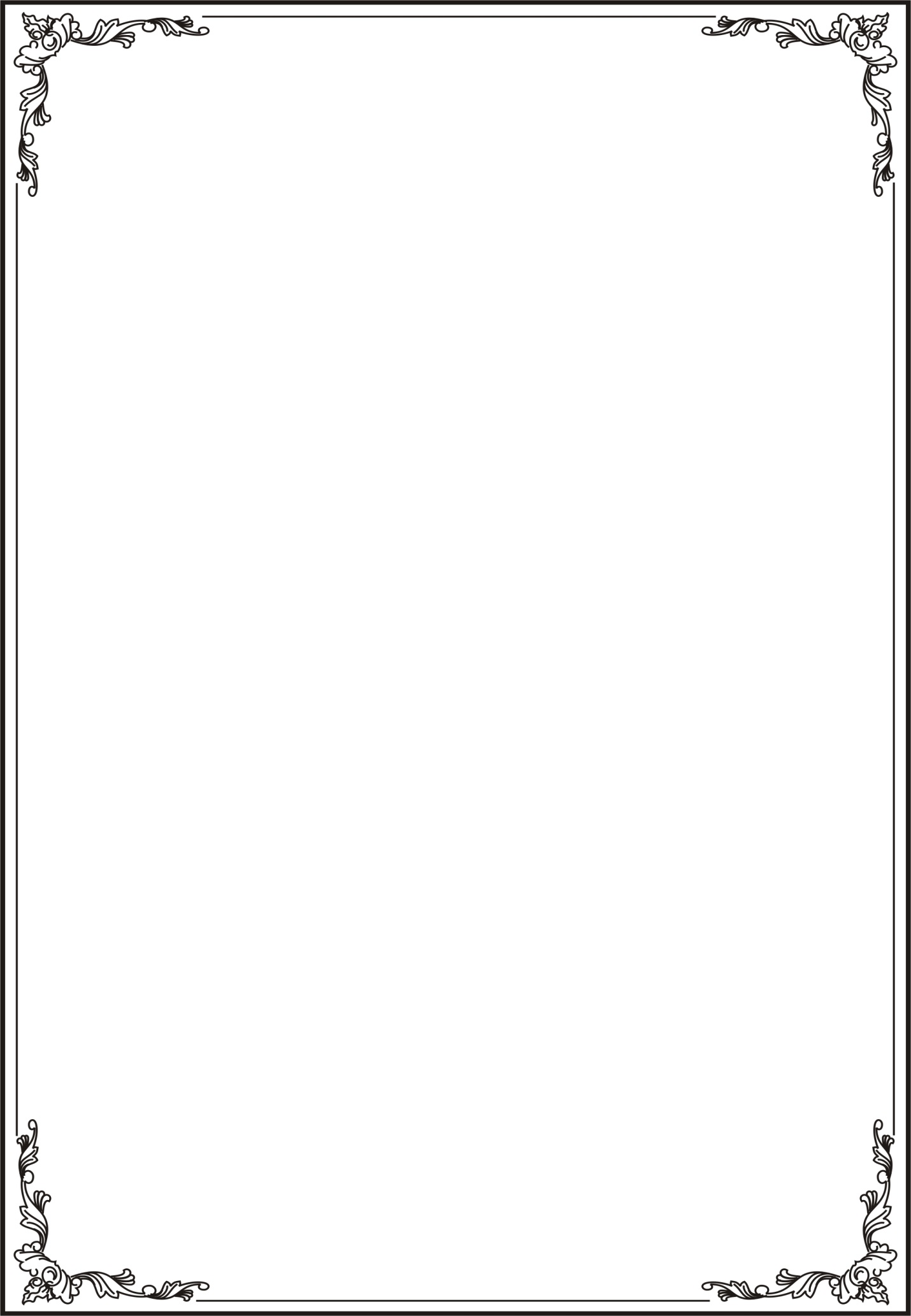
****

**BỘ GIÁO DỤC VÀ ĐÀO TẠO  
TRƯỜNG ĐẠI HỌC SƯ PHẠM KỸ THUẬT TP. HỒ CHÍ MINH  
KHOA ĐÀO TẠO CHẤT LƯỢNG CAO**---------------🙦 🕮 🙤---------------

*TP. Hồ Chí Minh, tháng 11, năm 2019*

**ỨNG DỤNG CÂY TÌM KIẾM NHỊ PHÂN**

**XÂY DỰNG TỪ ĐIỂN ANH –VIỆT**

***Đề tài:***

**CẤU TRÚC DỮ LIỆU VÀ GIẢI THUẬT**

***Giảng viên hướng dẫn:* Trần Công Tú**

***Lớp:* ST7**

***Sinh viên:***

**Lê Nhật Tường 18110234**

**Đinh Bách Thông 18110207**

**BÁO CÁO ĐỒ ÁN MÔN HỌC**

# MỤC LỤC

[MỤC LỤC 1](#_Toc25393566)

[DANH MỤC CÁC HÌNH 2](#_Toc25393567)

[NỘI DUNG 3](#_Toc25393568)

[I. Giới thiệu đề tài 3](#_Toc25393569)

[1. Lý do chọn đề tài 3](#_Toc25393570)

[2. Mục tiêu đề tài 3](#_Toc25393571)

[3. Cấu trúc dữ liệu và giải thật 3](#_Toc25393572)

[II. Quá trình thực hiện 4](#_Toc25393573)

[1. Thiết kế giao diện 4](#_Toc25393574)

[1.1. Form tìm kiếm 4](#_Toc25393575)

[1.2. Form cập nhật 6](#_Toc25393576)

[1.3. Form thêm từ mới 7](#_Toc25393577)

[1.4. Form xóa từ 9](#_Toc25393578)

[2. Cấu trúc dữ liệu và giải thuật 9](#_Toc25393579)

[2.1. Cây nhị phân tìm kiếm và ứng dụng làm từ điển 9](#_Toc25393580)

[2.1.1. Cây nhị phân tìm kiếm 9](#_Toc25393581)

[2.1.2. Ứng dụng và cách lưu trữ từ trên cây 10](#_Toc25393582)

[2.2. Các phương thức hỗ trợ trên cây nhị phân tìm kiếm 11](#_Toc25393583)

[2.3. Các giải thuật chính được sử dụng 13](#_Toc25393584)

[2.3.1. Chèn 1 nút vào cây nhị phân tìm kiếm 13](#_Toc25393585)

[2.3.2. Tìm kiếm trên cây nhị phân tìm kiếm 15](#_Toc25393586)

[2.3.3. Hủy 1 nút trên cây nhị phân tìm kiếm 16](#_Toc25393587)

[3. Cài đặt và kiểm thử 18](#_Toc25393588)

[3.1. Tra từ 18](#_Toc25393589)

[3.2. Cập nhật từ 19](#_Toc25393590)

[3.3. Thêm từ mới 19](#_Toc25393591)

[3.4. Xóa từ 20](#_Toc25393592)

[III. Phân công công việc 21](#_Toc25393593)

[IV. Kết luận 21](#_Toc25393594)

[4.1. Đánh giá mức độ hoàn thành 21](#_Toc25393595)

[4.2. Các khó khăn 22](#_Toc25393596)

[4.3. Hướng phát triển 22](#_Toc25393597)

[TÀI LIỆU KHAM KHẢO 23](#_Toc25393598)

# DANH MỤC CÁC HÌNH

[Hình 1.1: Thiết kế Form tìm kiếm 4](#_Toc24924716)

[Hình 1.2: Thiết kế giao diện cập nhật 6](#_Toc24924717)

[Hình 1.3: Thiết kế giao diện thêm từ mới 7](#_Toc24924718)

[Hình 1.4: Thiết kế giao diện xóa từ 9](#_Toc24924719)

[Hình 3.1: Kiểm thử chức năng tìm kiếm 18](#_Toc24924720)

[Hình 3.2: Kiểm thử chức năng cập nhật 19](#_Toc24924721)

[Hình 3.3: Kiểm thử chức năng thêm từ mới 20](#_Toc24924722)

[Hình 3.4: Kiểm thử chức năng xóa từ 20](#_Toc24924723)

# NỘI DUNG

## I. Giới thiệu đề tài

1. Lý do chọn đề tài

Toàn cầu hóa đang là xu thế chung của thế giới ngày nay. Việt Nam hiện đang là một quốc gia đang phát triển, đang trong xu thế mở cửa, hội nhập với nền kinh tế toàn cầu hóa của thế giới. Một trong những công cụ quan trọng để cho một quốc gia có thể hội nhập vào nền kinh tế thế giới đó chính là trình độ ngoại ngữ của những người dân nước ấy. Đặc biệt là các thế hệ học sinh, sinh viên mầm non tương lai của đất nước việc học ngoại ngữ đối với họ là một điều rất quan trọng. Trong các ngôn ngữ chính trên thế giới hiện nay thì có thể nói tiếng Anh là phổ cập nhất.

Để học tốt tiếng Anh thì không gì bằng một quyển từ điển tốt. Xuất phát từ nhu cầu thực tiễn đó nhóm lập trình của nhóm thực hiện đã xây dựng một chương trình từ điển Anh-Việt.

Do thao tác chính trên phần mền từ điển là việc tra từ (tìm kiếm thông tin về từ tiếng Anh) chính vì thế nhóm đã chọn cây nhị phân tìm kiếm làm cấu trúc dữ liệu chính cho đề tài này để cho việc tìm kiếm từ trở nên nhanh hơn và dễ dàng hơn.

2. Mục tiêu đề tài

Phát triển phần mền từ điển với khoảng 30 000 từ tiếng Anh thông dụng. Mỗi từ tiếng Anh sẽ có các thông tin về loại từ, phát âm, loại từ và các nghĩa của từ.

Chương trình có một giao diện đơn giản, thân thiện cho phép người dùng dễ dàng sử dụng. Để cho việc tra tra cứu từ tiếng Anh một cách dễ dàng hơn. Ngoài việc tra cứu từ phần mền từ điển còn cho phép người dung cập nhật từ, bổ sung từ, thêm từ hoặc xóa từ tiếng Anh trong từ điển.

3. Cấu trúc dữ liệu và giải thật

Sử dụng cây nhị phân, các hàm của cây nhị phân tìm kiếm để cài đặt chương trình tra từ điển kết hợp một số chức năng về quản lý từ điển. Bộ dữ liệu gồm các từ thông dụng trong tiếng Anh. Chương trình sử dụng giao diện Winform, từ cần tra được nhập từ bàn phím.

Dữ liệu từ điển được lưu trữ trên file *“duLieuTuDien.txt”* chứa từ tiếng Anh, phát âm, loại từ và nghĩa của từ (để cho đảm cho cây từ điển được cân bằng thì file dữ liệu vào không được sắp xếp theo ABC, mà được lưu theo thứ tự duyệt trước). Sau khi khởi động chương trình dữ liệu trên file được lưu trữ dưới cấu trúc cây nhị phân tìm kiếm.

Tổ chức từ điển theo cấu trúc cây nhị phân tìm kiếm. Mỗi từ gồm có các thông tin: Từ tiếng Anh, phiên âm, loại từ, các nghĩa Việt (theo cấu trúc danh sách tuyến tính, tối đa có 5 nghĩa trên 1 từ). Chương trình có các chức năng: nhập từ mới, hiệu chỉnh từ, xóa từ, tra từ, ghi từ vào file và load từ vào cây nhị phân tìm kiếm.

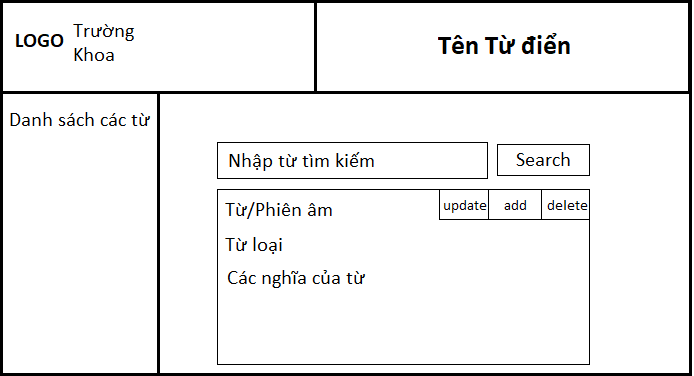
## II. Quá trình thực hiện

1. Thiết kế giao diện

Giao diện của từ điển T&T Dictionary được thiết kế đơn giản, thân thiện để người dùng dễ dàng sử dụng. Phần giao diện này được tổ chức thành 4 Form nhằm thực hiện các chức năng riêng: tra từ, tạo từ mới, cập nhật từ đã có và xóa từ trong file.

1.1. Form tìm kiếm

Giao diện tìm kiếm là giao diện chính của từ điển dùng để nhập từ cần tra, tra từ và chuyển sang các giao diện chức năng khác của từ điển.



Hình 1.1: Thiết kế Form tìm kiếm

Để làm được giao diện này, khi thiết kế Form1 của chương trình, nhóm cần thêm sử dụng các thành phần sau:

Thành phần phục vụ cho chức năng tra từ:

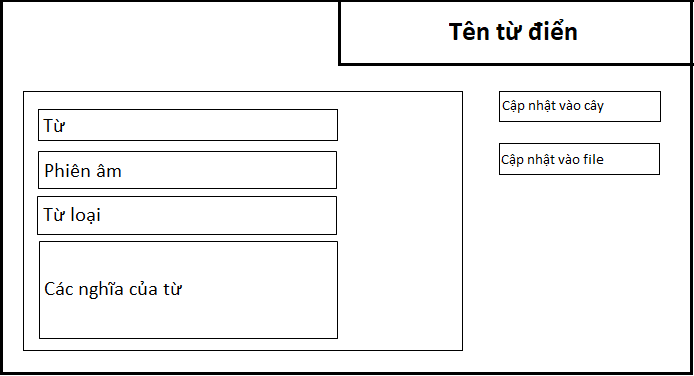
|  |  |  |  |
| --- | --- | --- | --- |
| Thành phần | Name | Text | Mô tả |
| TextBox | **tbx\_Nhaptu** |  | Dùng để nhập từ cần tra vào |
| Button | **btn\_Search** | Search | Sau khi click chuột vào **btn\_Search**, chương trình sẽ thực hiện thao tác tìm kiếm |
| Label | **lb\_Word** | Từ tiếng Anh | **lb\_Word** chứa từ và phần phiên âm của từ |
| **lb\_WordType** | Loại từ | **lb\_WordType** chứa thông tin về từ loại |
| **lb\_Mean1** | nghia 1 | Các label **lb\_Mean1**, **lb\_Mean2**, **lb\_Mean3**, **lb\_Mean4**, **lb\_Mean5** chứa các nghĩa của từ. Có tối đa là 5 nghĩa sẽ lần lượt được thể hiện ở các label trên. |
| **lb\_Mean2** | nghia 2 |
| **lb\_Mean3** | nghia 3 |
| **lb\_Mean4** | nghia 4 |
| **lb\_Mean5** | nghia 5 |

Thành phần tiếp theo có nhiệm vụ chuyển sang các chức khác của từ điển:

|  |  |  |  |
| --- | --- | --- | --- |
| Thành phần | Name | Text | Mô tả |
| Button | **btn\_Update** | Update | Sau khi click vào button này, chương trình sẽ load Form2 để thực hiện chức năng cập nhật từ |
| **btn\_Add** | Add New | Sau khi click vào button này, chương trình sẽ load Form3 để thực hiện chức năng thêm từ mới |
| **btn\_Delete** | Delete | Sau khi click vào button này, từ bạn vừa tra sẽ bị xóa khỏi cây nhị phân tìm kiếm và load Form4 cho phép người dùng xóa hẳn từ đó trong file (nếu muốn) |

1.2. Form cập nhật

Sau khi người dùng thực hiện tao tác tra từ, các thông tin về từ như phiên âm, từ loại, nghĩa sẽ được hiện ra. Nếu người dùng phát hiện ra lỗi sai hay thiếu sót trong thông tin của từ và muốn cập nhật lại thì giao diện cập nhật sẽ hỗ trợ người dùng thực hiện chức năng này.



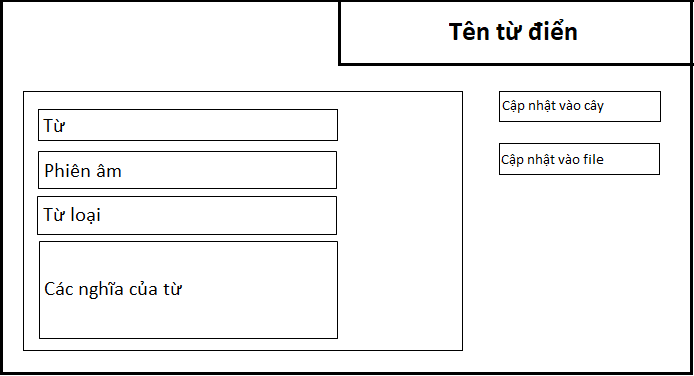
Hình 1.2: Thiết kế giao diện cập nhật

Trong chương trình, Form2 chính là giao diện cập nhậtcủa từ điển. Form2 gồm các thành phần sau:

|  |  |  |  |
| --- | --- | --- | --- |
| Thành phần | Name | Text | Mô tả |
| TextBox | **tbx\_word** |  | **tbx\_word** dùng để chứa từ |
| **tbx\_pronunciation** |  | **tbx\_pronunciation** dùng để chứa phiên âm của từ |
| **tbx\_wordType** |  | **tbx\_wordType** chứa thông tin về từ loại |
| **tbx\_mean1**  **tbx\_mean2**  **tbx\_mean3**  **tbx\_mean4**  **tbx\_mean5** |  | 5 textbox **tbx\_mean1**, **tbx\_mean2**, **tbx\_mean3**, **tbx\_mean4**, **tbx\_mean5** chứa các nghĩa của từ |
| Label | **lb\_word** | Từ: | Các label **lb\_word**, **lb\_pronunciation**, **lb\_wordType**, **lb\_mean** nằm ngang với các textbox giúp cho người dùng dễ dàng nhận biết chức năng của từng textbox trong form |
| **lb\_pronunciation** | Phát âm: |
| **lb\_wordType** | Từ loại: |
| **lb\_mean** | Nghĩa: |
| Button | **btn\_updateTree** | Cập nhật vào cây… | **btn\_updateTree** sẽ cho phép chương trình ghi nhận các cập nhật vào cây khi người dùng click vào button này |
| **btn\_updateFile** | Cập nhật vào file… | **btn\_updateFile** sẽ cập nhật tất cả thay đổi vào file *“dulieuTuDien.txt”* |

1.3. Form thêm từ mới

Trong quá trình sử dụng từ điển, khi người dùng phát hiện ra trong dữ liệu từ điển có thiết sót các từ và muốn thêm từ đó vào thì chức năng thêm từ mới ở giao diện thêm từ sẽ hỗ trợ người dùng thực hiện công việc trên.



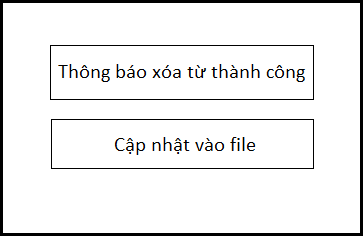
Hình 1.3: Thiết kế giao diện thêm từ mới

**Form3** của từ điển gồm các thành phần tương tự như ở Form2, nhưng thay vì các text trong TextBox lấy thông tin của từ sẵn có thì các text ở Form3 để trống cho người dùng thêm vào.

|  |  |  |  |
| --- | --- | --- | --- |
| Thành phần | Name | Text | Mô tả |
| TextBox | **tbx\_word** |  | **tbx\_word** dùng để chứa từ |
| **tbx\_pronunciation** |  | **tbx\_pronunciation** dùng để chứa phiên âm của từ |
| **tbx\_wordType** |  | **tbx\_wordType** chứa thông tin về từ loại |
| **tbx\_mean1**  **tbx\_mean2**  **tbx\_mean3**  **tbx\_mean4**  **tbx\_mean5** |  | 5 textbox **tbx\_mean1**, **tbx\_mean2**, **tbx\_mean3**, **tbx\_mean4**, **tbx\_mean5** chứa các nghĩa của từ |
| Label | **lb\_word** | Từ: | Các label **lb\_word**, **lb\_pronunciation**, **lb\_wordType**, **lb\_mean** nằm ngang với các textbox giúp cho người dùng dễ dàng nhận biết chức năng của từng textbox trong form |
| **lb\_pronunciation** | Phát âm: |  |
| **lb\_wordType** | Từ loại: |  |
| **lb\_mean** | Nghĩa: |  |
| Button | **btn\_updateTree** | Cập nhật vào cây… | **btn\_updateTree** sẽ cho phép chương trình ghi nhận các cập nhật vào cây khi người dùng click vào button này |
| **btn\_updateFile** | Cập nhật vào file… | **btn\_updateFile** sẽ cập nhật tất cả thay đổi vào file *“dulieuTuDien.txt”* |

1.4. Form xóa từ

Sau khi click chuột vào button **btn\_Delete** ở Form1, nếu từ bị xóa hợp lệ thì ngay lập tức giao diện xóa từ sẽ được load mới mục đích thông báo xóa từ thành công và cung cấp thêm chức năng xóa hẳn từ trên file *“dulieuTuDien.txt”.*

**

Hình 1.4: Thiết kế giao diện xóa từ

|  |  |  |  |
| --- | --- | --- | --- |
| Thành phần | Name | Text | Mô tả |
| Label | **label1** | Xóa từ thành công | **label1** nhằm thông báo cho người dùng biết hiện tại từ đã được xóa khỏi cây nhị phân tìm kiếm |
| Button | **button1** | Nhấn để cập nhật vào file | Nếu click chuột vào **button1** thì từ vừa bị xóa khỏi cây sẽ mất hẳn trên file *“dulieuTuDien.txt”* |

2. Cấu trúc dữ liệu và giải thuật

2.1. Cây nhị phân tìm kiếm và ứng dụng làm từ điển

2.1.1. Cây nhị phân tìm kiếm

Một cây tìm kiếm nhị phân (Binary Search Tree – viết tắt là BST) là một cây mà trong đó tất cả các nút đều có các đặc điểm sau:

* Cây con bên trái của một nút có khóa (key) nhỏ hơn giá trị khóa của nút cha (của cây con này).
* Cây con bên phải của một nút có khóa lớn hơn giá trị khóa của nút cha (của cây con này).

Một cây rỗng có thể coi là cây nhị phân tìm kiếm. Dựa vào định nghĩa chúng ta có một số nhận xét như sau:

* Trên cây nhịp phân tìm kiếm, không có các nút cùng khóa.
* Các cây con trái, phải của một cây nhị phân tìm kiếm cũng là một cây nhị phân tìm kiếm.

2.1.2. Ứng dụng và cách lưu trữ từ trên cây

***Ứng dụng:***

Công dụng chính của cây nhị phân tìm kiếm là việc tìm kiếm các Node trên cây một cách dễ dàng và tiết kiệm thời gian. Điều này rất phù hợp với thao tác tra từ (thao tác chính trên từ điển). Kèm theo các thao tác trên cây nhị phân tìm kiếm: như thêm một nút vào cây, xóa một nút trên cây…, các thao tác này cũng rất phù hợp với các thao tác cơ bản của phần mềm từ điển. Chính vì thế cấu trúc cây nhị phân tìm kiếm đã được ứng dụng vào chương trình một cách hiệu quả và triệt để.

***Cách tổ chức từ trên file:***

Dữ liệu từ điển được lưu trên file “duLieuTuDien.txt”. Mỗi từ được lưu trữ trên 4 dòng:

* Dòng 1: Chứa thông tin về từ tiếng Anh.
* Dòng 2: Mang thông tin về phiên âm quốc tế của từ.
* Dòng 3: Chứa thông tin về loại từ.
* Dòng 4: gồm các nghĩa của từ, các nghĩa được phân cách với nhau bởi ký tự ‘/’.

***Cách lưu trữ từ trên cây:***

Sử dụng phần dữ liệu trong một nút của cây để lưu trữ các công tin về từ. Một Node của cây nhị phân tìm kiếm có kiểu dữ liệu aboutWord chứa tất cả các thông tin về một từ bao gồm từ tiếng Anh, phiên âm quốc tế, loại từ và các nghĩa của từ. Việc tìm kiếm từ là tìm kiếm một Node chứa từ ta muốn tìm.

*Kiểu dữ liệu aboutWord chứa tất cả các thông tin về từ.*

public struct aboutWord

{

public string word;

public string pronunciation;

public string wordType;

public string mean;

}

*Khai báo node.*

public class Node  
{  
 public aboutWord data;  
 public Node pLeft;  
 public Node pRight;  
}

*Cách đọc file.*

|  |  |
| --- | --- |
| **Giải thích thuật toán:** | Đọc dữ liệu từ file “*duLieuTuDien.txt”*. Lấy lần lượt thông tin từng dòng và đưa chúng vào các thành phần của struct aboutWord: word, pronunciation, wordType, mean. |
| **Code trên C#** | FileStream fs = new FileStream(@"duLieuTuDien.txt", FileMode.Open);  StreamReader rd = new StreamReader(fs, Encoding.UTF8);  while ((aboutWord1.word = rd.ReadLine()) != null)  {  aboutWord1.pronunciation = rd.ReadLine();  aboutWord1.wordType = rd.ReadLine();  aboutWord1.mean = rd.ReadLine();  tree.root = tree.insert(tree.root, aboutWord1);  }  rd.Close(); |

2.2. Các phương thức hỗ trợ trên cây nhị phân tìm kiếm

Các phương thức hỗ trợ trên cây nhị phân tìm kiếm bao gồm: khai báo Node, tạo cây rỗng, kiểm tra cây rỗng, các thủ tục duyệt cây.

Các phương thức khai báo Node, tạo cây rỗng, kiểm tra cây rỗng được sử dụng trong quá trình khởi tạo cây. Thủ tục duyệt cây NLR được ứng dụng khi cập nhật các từ vào file *“duLieuTuDien.txt”.*

|  |  |
| --- | --- |
| **Thuật toán** | **Code trên C#** |
| ***Khai báo node***  Mội Node bao gồm khóa data chứa tất cả thông tin liên quan đến từ, một Node pLeft biểu thị cho cây con trái và một node pRight biểu chị cho cây con phải. | public class Node  {  public aboutWord data;  public Node pLeft;  public Node pRight;  public int height;  //Tạo một node mới với khóa là Key, không nhánh  public Node(aboutWord key)  {  data = key;  height = 1;  pLeft = pRight = null;  }  } |
| ***Tạo cây rỗng***  Tạo cây rỗng với giá trị nút gốc bằng NULL. | public class TREE  {  public Node root;  //Khởi tạo cây rỗng  public TREE()  {  root = null;  }  } |
| ***Kiểm tra cây rỗng***  Một cây là rổng khi và chỉ khi nút gốc có giá trị bằng NULL. | int isEmptyTree(Node root)  {  if (root == null)  return 0;  return 1;  } |
| ***Kiểm tra nút lá***  Một nút gọi là lá khi không có nút con nào, tức là giá trị pLeft và pRight là NULL. | int isLeafNode(Node root)  {  if (root.pLeft == null && root.pRight == null)  return 1;  return 0;  } |
| ***Duyệt NLR***  Sử dụng thủ tục duyệt cây NLR để cập nhật dữ liệu từ cây nhị phân tìm kiếm vào File “duLieuTuDien.txt”. | public void duyet\_NLR(Node node1, StreamWriter wr)  {  if (node1 == null) return;  wr.WriteLine(node1.data.word);  wr.WriteLine(node1.data.pronunciation);  wr.WriteLine(node1.data.wordType);  wr.WriteLine(node1.data.mean);  duyet\_NLR(node1.pLeft, wr);  duyet\_NLR(node1.pRight, wr);  }  public void writeToFile(Node node)  {  StreamWriter wr = new StreamWriter(@"duLieuTuDien.txt", false, Encoding.UTF8);  duyet\_NLR(node, wr);  wr.Close();  } |

2.3. Các giải thuật chính được sử dụng

2.3.1. Chèn 1 nút vào cây nhị phân tìm kiếm

Việc chèn một khóa trên cây nhị phân tìm kiếm có thể thực hiện nhờ đệ quy.

Ứng dụng giải thuật thêm một nút vào cây nhị phân tìm kiếm để thêm một từ mới vào cây nhị phân. Một nút mới được thêm vào có kiểu dữ kiệu là aboutWord (lưu trữ các thông tin về từng vựng).

|  |  |
| --- | --- |
| **Giải thích thuật toán** | ***Chèn 1 nút vào cây nhị phân tìm kiếm***  Chúng ta bắt đầu từ gốc. Nếu khóa của gốc khác khóa cần chèn ta tìm nó trong cây con trái hoặc phải. Nếu cây con trái hoặc phải tương ứng là rỗng (không tìm thấy) thì thêm một nút và gán cho nút ấy khóa cần chèn. Hàm sẽ trả về giá trị 0 khi chúng ta thêm 1 từ đã có trong cây, trả về giá trị 1 khi việc thêm nút vào cây hoàn tất và trả về giá trị -1 khi không cấp phát đủ bộ nhớ cho nút thêm vào. |
| **Code trên C#** | public int InsertNode(ref Node T, aboutWord key)  {  if (T != null)  {  if (String.Compare(key.word, T.data.word, true) == 0)  return 0;  else if (String.Compare(key.word, T.data.word, true) < 0)  return InsertNode(ref T.pLeft, key);  else  return InsertNode(ref T.pRight, key);  }  Node temp = new Node(key);  if (temp == null)  return -1;  T = temp;  return 1;  } |
| **Giải thích thuật toán** | Đối với cây nhị phân tìm kiếm cân bằng, sau khi thực hiện chèn một nút vào cây, ta kiểm tra xem cây có bị mất cân bằng hay không. Nếu có, ta thực hiện thao tác xoay trái hoặc xoay phải cây (tùy theo trường hợp mất cân bằng) để đưa cây về trạng thái cân bằng. |
| **Code trên C#** | ***Code mẫu do nhóm lập trình kham khảo và vận dụng thành công vào đồ án từ điển.***  public Node insert(Node node, aboutWord key)  {  if (node == null)  return (new Node(key));  if (String.Compare(key.word, node.data.word, true) < 0)  node.pLeft = insert(node.pLeft, key);  else if (String.Compare(key.word, node.data.word, true) > 0)  node.pRight = insert(node.pRight, key);  else // Duplicate keys not allowed  return node;  node.height = 1 + max(height(node.pLeft),  height(node.pRight));  int balance = getBalance(node);  if (balance > 1 && string.Compare(key.word, node.pLeft.data.word) < 0)  return rightRotate(node);  // Right Right Case  if (balance < -1 && string.Compare(key.word, node.pRight.data.word) > 0)  return leftRotate(node);  // Left Right Case  if (balance > 1 && string.Compare(key.word, node.pLeft.data.word) > 0)  {  node.pLeft = leftRotate(node.pLeft);  return rightRotate(node);  }  // Right Left Case  if (balance < -1 && string.Compare(key.word, node.pRight.data.word) < 0)  {  node.pRight = rightRotate(node.pRight);  return leftRotate(node);  }  /\* return the (unchanged) node pointer \*/  return node;  } |

2.3.2. Tìm kiếm trên cây nhị phân tìm kiếm

Tìm kiếm thông tin một từ trong từ điển được ứng dụng từ giải thuật tìm kiếm một nút trên cây nhị phân tìm kiếm. Khi chúng ta muốn tìm kiếm một từ hàm sẽ trả giá trị về nút đang chứa thông tin của từ đó. Việc hiển thị thông tin của từ được truy xuất từ nút chúng ta mới tìm được.

Việc tìm kiếm cũng tương tự như việc thêm phần tử vào cây nhị phân tìm kiếm. Ta có quy trình như sau:

|  |  |
| --- | --- |
| **Giải thích thuật toán** | ***Tìm kiếm trên cây nhị phân tìm kiếm***  - Gán giá trị nút gốc cho nút p  - Nếu Node hiện tại có giá trị bằng giá trị cần tìm, trả về node p và kết thúc.  - Nếu Node hiện tại có giá trị lớn hơn giá trị cần tìm, ta gán giá trị p.Pleft cho p để tiếp tục tìm ở cây con bên trái.  - Nếu Node hiện tại có giá trị nhỏ hơn giá trị cần tìm, ta gán giá trị p.Right cho p để tiếp tục tìm ở cây con bên phải.   * Nếu tìm đến hết cây (Node đó = NULL) mà không tìm thấy nút nào có khóa cần tìm, trả về giá trị NULL cho nút và kết thúc. |
| **Code trên C#** | Tìm kiếm 1 nút trên cây có khóa là key.  public Node Search(Node node, string key)  {  Node p = node;  while(p != null)  {  if (string.Compare(key, p.data.word) == 0)  return p;  else if (string.Compare(key, p.data.word) < 0)  p = p.pLeft;  else  p = p.pRight;  }  return null;  } |

2.3.3. Hủy 1 nút trên cây nhị phân tìm kiếm

Khi thực hiện xóa 1 phần tử x khỏi cây, phải đảm bảo ràng buộc của cây nhị phân tìm kiếm. Xảy ra 3 trường hợp sau:

* Trường hợp x là nút lá: với trường hợp này ta chỉ việc giải phóng bộ nhớ cho phần tử này mà thôi.
* Trường hợp x có 1 nút con: khi đó chúng ta thực hiện 2 việc.

+ Móc nối nút cha của x với con duy nhất của x.

+ Sau đó hủy phần tử x

* Trường hợp x có 2 nút con: trong trường hợp này chúng ta không thể hủy phần tử x mà phải thay nó bằng nút lớn nhất trên cây con trái hoặc nút nhỏ nhất trên cây con phải. Khi đó nút được giải phóng bộ nhớ là một trong 2 nút này.

Ứng dụng giải thuật xóa một nút trên cây nhị phân tìm kiếm để xóa một từ trong từ điển. Việc xóa một từ chính là tìm một nút chứa thông tin của từ đó và xóa nút này ra khỏi cây nhị phân tìm kiếm.

|  |  |
| --- | --- |
| **Giải thích thuật toán** | ***Hàm xóa node có khóa là key***  Trong thuật toán này ta xem việc xóa nút trong trường hợp là nút lá hoặc có 1 nút con là như nhau. Trong trường hợp cây có 1 nút con thì nút cha của x sẽ nối với con duy nhất của x, còn nếu x là nút lá thì nói nút cha của x với 1 trong 2 nút con của x đều có giá trị là null. Trường hợp nút có 2 con, chúng ta sẽ thay thế x bằng nút nhỏ nhất bên cây con phải (là phần tử cực trái trên cây con phải). |
| **Code trên C#** | ***Hàm xóa node có khóa là key***  public void DeleteNode(ref Node node, string key)  {  if (node == null)  return;  else  {  if (String.Compare(key, node.data.word, true) < 0)  DeleteNode(ref node.pLeft, key);  else if (String.Compare(key, node.data.word, true) > 0)  DeleteNode(ref node.pRight, key);  else  {  Node X = node;  if (node.pLeft == null)  node = node.pRight;  else if (node.pRight == null)  node = node.pLeft;  else  {  DiTimNodeTheMang(ref X, ref node.pRight);  }  X = null;  }  }  }  ***Đi tìm node thế mạng là nút nhỏ nhất trên cây con phải.***  public void DiTimNodeTheMang(ref Node X, ref Node Y)  {  if (Y.pLeft != null)  DiTimNodeTheMang(ref X, ref Y.pLeft);  else  {  X.data = Y.data;  X = Y;  Y = Y.pRight;  }  } |

3. Cài đặt và kiểm thử

Dữ liệu từ điển hiện tại có khoảng 100 000 từ được lưu trữ trong file *“duLieuTuDien.txt”*. Mỗi khi chạy chương trình, dữ liệu này sẽ được đọc và thêm vào cây để thực hiện các thao tác trên cây nhị phân tìm kiếm.

Trong quá trình kiểm thử chương trình, nhóm thực hiện đã kiểm tra từng chức năng của từ điển với các trường hợp khác nhau. Cụ thể quá trình kiểm thử từng chức năng được mô tả sau đây.

3.1. Tra từ

Đầu tiên, nhập từ cần tra vào textbox **tbx\_Nhaptu** rồi click chuột vào button **btn\_Search**. Ngay lập tức thao tác tìm kiếm sẽ được thực hiện. Khi từ được tìm ra **lb\_Word** sẽ hiển thị từ và phần phiên âm của từ. **lb\_WordType** hiển thị từ loại của từ. Các lable **lb\_Mean1**, **lb\_Mean2**, **lb\_Mean3**, **lb\_Mean4**, **lb\_Mean5** sẽ hiện thị từ nghĩa thứ nhất đến nghĩa thứ 5 của từ vừa tra. Nếu từ không có đủ 5 nghĩa chỉ hiển thị đủ số nghĩa của từ.



Hình 3.1: Kiểm thử chức năng tìm kiếm

Trường hợp chưa nhập từ vào textbox **tbx\_Nhaptu**, màn hình sẽ xuất hiện MessageBox thông báo “Bạn phải nhập từ trước khi tìm kiếm”.

Trường hợp từ vừa nhập không có trong dữ liệu của cây nhị phân tìm kiếm thì text của **lb\_Word** sẽ hiển thị “Xin lỗi! Không có dữ liệu từ này”

3.2. Cập nhật từ

Đầu tiên thực hiện thao tác tìm kiếm từ “fox” trong từ điển, kết quả cho ra từ “fox” có 1 nghĩa là “con cáo”.

Click chuột vào **btn\_Update** thì chương trình chuyển sang Form2 chứa thông tin của từ trong các TextBox. Nhập thêm 1 nghĩa là “người ranh ma” cho từ và click chuột vào button **btn\_updateTree** như hình bên dưới:

****

Hình 3.2: Kiểm thử chức năng cập nhật

3.3. Thêm từ mới

Ta thực hiện tìm kiếm từ “apple” trong từ điển

Do trong từ điển không có dữ liệu của từ này nên Form1 không thể cho ra thông tin của từ. Nếu muốn thêm từ này vào dữ liệu của từ điển, ta click chuột vào **btn\_Add** để load Form3 hỗ trợ cho thao tác thêm từ mới.

Nhập tất cả thông tin về từ “apple” vào Form3



Hình 3.3: Kiểm thử chức năng thêm từ mới

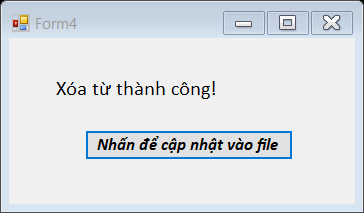
Sau khi nhập đầy đủ thông tin của từ, click chuột vào **btn\_updateTree** để thêm từ vừa nhập vào cây. Nếu thao tác hợp lệ thì màn hình xuất hiện MessageBox thông báo thêm từ thành công.

Nếu người dùng muốn thêm dữ liệu của từ “apple” vĩnh viễn vào file *“dulieuTuDien.txt”* thì cần click vào button **btn\_updateFile**. Lúc đó MessageBox thông báo thêm từ vào file thành công xuất hiện và dữ liệu của từ đã có ở trong file.

3.4. Xóa từ

Muốn xóa 1 từ ra khỏi dữ liệu của từ điển, trước tiên ta cần tìm kiếm thông tin của từ đó. Ví dụ ta tìm kiếm từ “woodcutter”

Để xóa từ “woodcutter” ra khỏi từ điển, ta click bào button **btn\_Delete**. Nếu xóa hợp lệ thì Form4 sẽ được load để thông báo xóa thành công và cho người dùng lựa chọn xóa từvĩnh viễn khỏi file *“dulieuTuDien.txt”* bằng cách click chuột vào **button1** (nếu muốn xóa) hoặc không.



Hình 3.4: Kiểm thử chức năng xóa từ

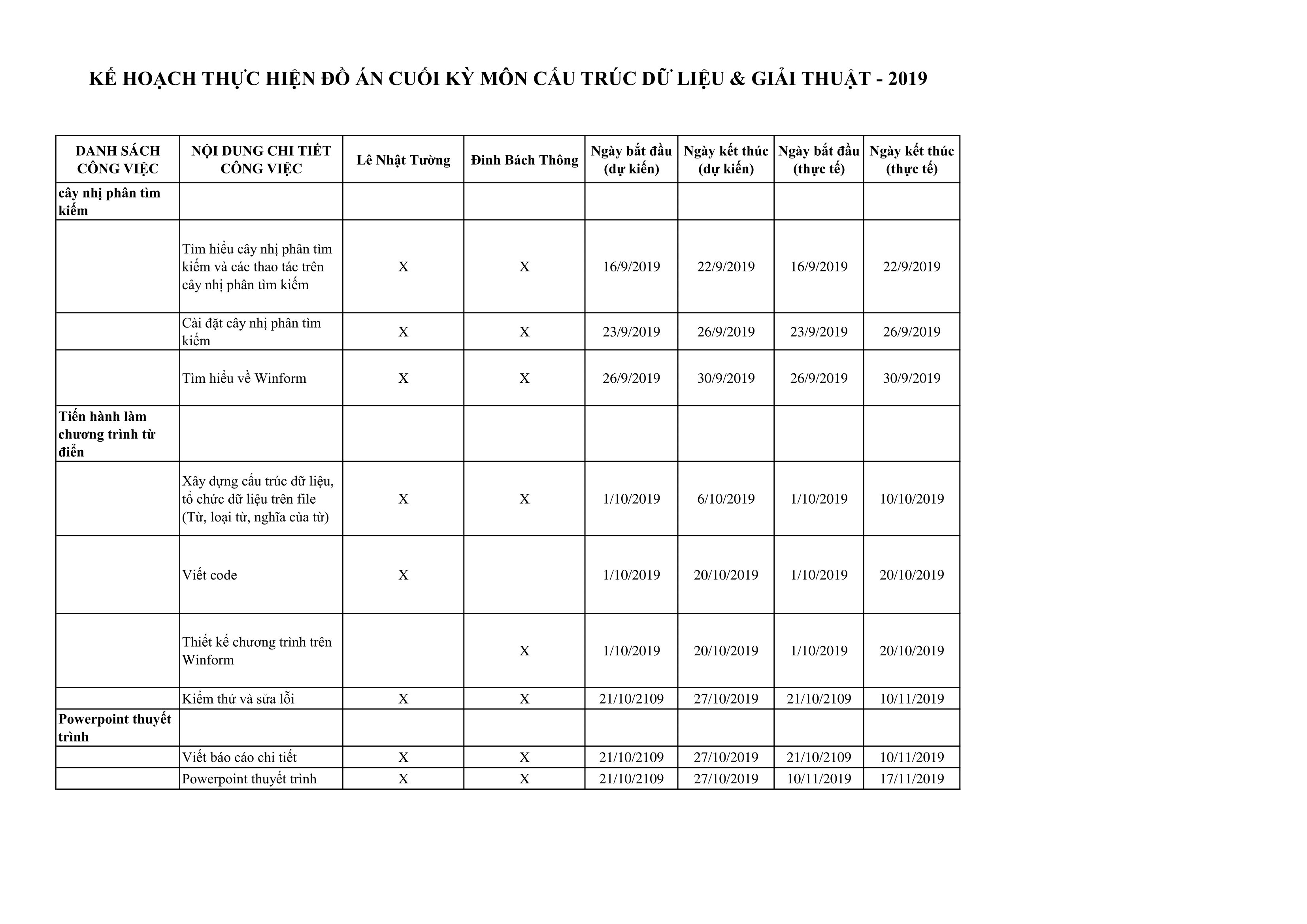
Trường hợp chưa nhập từ vào textbox **tbx\_Nhaptu** mà đã thực hiện xóa thì màn hình sẽ hiện ra MessageBox thông báo “Nhập từ cần tìm trước khi xóa”

## III. Phân công công việc

***Yêu cầu bài toán:***

Tổ chức từ đển theo cấu trúc cây nhị phân, mỗi từ gồm có các thông tin: Từ, loại từ, các nghĩa việt (theo cấu trúc danh sách tuyến tính). Chương trình có các chức năng: nhập từ mới, hiệu chỉnh từ, xóa từ; ghi từ vào file, load từ vào danh sách liên kết…

Lưu ý: - Danh sách các từ được tổ chức theo cấu trúc cây nhị phân tìm kiếm.



## IV. Kết luận

4.1. Đánh giá mức độ hoàn thành

Nhóm thực hiện tự đánh giá rằng nhóm đã cơ bản hoàn thành các yêu cầu của đồ án. Trong đồ án này, nhóm thực hiện đã tìm hiểu được các kiến thức về cây nhị phân tìm kiếm, cách tổ chức dữ liệu trên cây và các thao tác trên cây như tìm kiếm, thêm và xóa nút trên cây. Áp dụng vào sản phẩm từ điển để thực hiện các chức năng tra từ, chỉnh sửa, thêm từ mới và xóa từ. Nhìn chung các thuật toán được áp dụng một cách linh hoạt để có thể hoàn thành ứng dụng từ điển này.

Tuy nhiên, sản phẩm vẫn chỉ là một ứng dụng mô phỏng các thuật toán trên cây nhị phân. Để có thể đem sản phẩm vào thực tiễn để sử dụng, nhóm thực hiện cần một quá trình để cải tiến và phát triển thêm.

4.2. Các khó khăn

Trong quá trình thực hiện đồ án, nhóm đã gặp không ít những khó khăn. Bởi vì đây là lần đầu tiên lập trình một ứng dụng, nhóm phải bắt đầu tìm hiểu tất cả từ con số 0. Từ tìm kiếm các tài liệu về cây nhị phân tìm kiếm đến việc áp dụng cấu trúc này trên một ngôn ngữ lập trình mới là C#. Bên cạnh đó, áp lực về thời gian và lượng công việc là những khó khăn mà nhóm đã gặ phải. Tuy nhiên nhóm thực hiện đã vượt qua tất cả khó khăn để hoàn thành sản phẩm bằng tất cả khả năng của mình.

4.3. Hướng phát triển

Thời gian tiếp theo, nhóm thực hiện sẽ tiếp tục cải thiện và phát triển một số tính năng mới cho từ điển để có thể đem vào thực tế để sử dụng.

Nhóm đã tìm hiểu về cây AVL và dựa vào một số tài liệu để áp dụng cây nhị phân cân bằng vào từ điển. Tuy nhiên nhóm chưa thực sự nắm rõ các thao tác về cây nhị phân cân bằng. Trong thời gian tới, nhóm sẽ tiếp tục nghiên cứu tiếp để có thể nắm rõ hơn và vận dụng một cách hiệu quả cây AVL vào từ điển

Thứ hai, nhóm thực hiện sẽ tìm cách để sử dụng các dữ liệu từ điển từ bên ngoài. Như vậy, số lượng từ trong dữ liệu từ điển mới đủ đáp ứng nhu cầu tìm kiếm của thực tế.

Kế tiếp, nhóm thực hiện sẽ phát triển thêm tính năng từ điển nói. Từ điển sẽ cung cấp cho người dùng nghe được cách đọc của những từ mà họ tìm kiếm. Tính năng này hỗ trợ cho nhu cầu học tiếng Anh của người dùng một cách tốt hơn khi có thể tra được không chỉ thông tin của từ mà còn cách đọc của từ đó.

Cuối cùng, nhóm thực hiện sẽ phát triển thêm từ điển Việt-Anh để có thể hoàn thiện hơn sản phẩm của mình.

# TÀI LIỆU KHAM KHẢO

[1] Thien Tam Nguyen, Cây Nhị Phân Tìm Kiếm (Cấu Trúc Dữ Liệu Và Giải Thuật),\_https://www.youtube.com/playlist?list=PLimFJKGsbn1n6UkQnTjXhziglRzmJ2mlR&fbclid=IwAR1yNAJVDngViMR7bhh68Ommwyeh8MyF61z0eLwRFEA8GB9exbqDRPJ9d4Y

[2] AVL Tree set 1, GreeksforGreek, https://www.geeksforgeeks.org/avl-tree-set-1-insertion/

[3] Lê Văn Vinh, Giáo trình Cấu trúc dữ liệu và giải thuật

[4] Kho từ vựng tiếng anh, https://daynhauhoc.com/t/xin-file-tu-dien-anh-viet-viet-anh-de-lam-tu-dien/15532