MỤC LỤC

[DANH MỤC HÌNH ẢNH 2](#_Toc27054064)

[MỞ ĐẦU 3](#_Toc27054065)

[1. Lý do chọn đề tài 3](#_Toc27054066)

[2. Mục tiêu cần đạt được 3](#_Toc27054067)

[3. Đối tượng và phạm vi nghiên cứu 3](#_Toc27054068)

[CHƯƠNG I:](#_Toc27054069) [TÌM HIỂU VỀ BẢNG BĂM VÀ KỸ THUẬT BĂM 4](#_Toc27054070)

[1. Bảng băm là gì? 4](#_Toc27054071)

[2. Lý thuyết về hàm băm 5](#_Toc27054072)

[Phương pháp phân đoạn 6](#_Toc27054073)

[3. Giải quyết va chạm 7](#_Toc27054074)

[3.1 Va chạm là gì? 7](#_Toc27054075)

[3.2 Cách giải quyết va chạm 8](#_Toc27054076)

[CHƯƠNG II: BÀI TOÁN ỨNG DỤNG 11](#_Toc27054077)

[1. Nội dung 11](#_Toc27054078)

[1.1 File đầu vào 11](#_Toc27054079)

[1.2 Giới thiệu giao diện 12](#_Toc27054080)

[2. Chương trình minh họa 13](#_Toc27054081)

[2.1 Hàm băm 13](#_Toc27054082)

[2.2 Các chức năng của từ điển 14](#_Toc27054083)

[KẾT LUẬN 22](#_Toc27054084)

[1. Kết quả đạt được 22](#_Toc27054085)

[2. Những khó khăn 22](#_Toc27054086)

[3. Những khuyết điểm 22](#_Toc27054087)

[TÀI LIỆU THAM KHẢO 24](#_Toc27054088)

# DANH MỤC HÌNH ẢNH

[Hình 1. 1 Va chạm trong bảng băm 8](file:///C:\Users\PC\Downloads\báo-cáo%20(1).docx#_Toc27054875)

[Hình 1. 2 Kỹ thuật phân tách chuỗi 10](#_Toc27054876)

[Hình 2. 1 File txt chứa thông tin đầu vào 8](file:///C:\Users\PC\Downloads\báo-cáo%20(1).docx#_Toc27054805)

[Hình 2. 2 Giao diện chính của từ điển 12](#_Toc27054806)

[Hình 2. 3 Tổng mã ACII 13](#_Toc27054807)

[Hình 2. 4 Lấy 3 giá trị cuối 13](#_Toc27054808)

[Hình 2. 5 Chức năng tìm từ 14](#_Toc27054809)

[Hình 2. 6 Khởi tạo danh sách tạm 14](#_Toc27054810)

[Hình 2. 7 “Băm” từ tiếng Anh thành số 15](#_Toc27054811)

[Hình 2. 8 Tiến hành tìm từ 15](#_Toc27054812)

[Hình 2. 9 Ấn vào nút “Add” để tiến hành thêm từ 16](#_Toc27054813)

[Hình 2. 10 Nhập thông tin vào 16](#_Toc27054814)

[Hình 2. 11 Thêm vào ngăn rỗng 17](#_Toc27054815)

[Hình 2. 12 Thêm từ khi ngăn đã tồn tại phần tử 17](#_Toc27054816)

[Hình 2. 13 Nhấn vào nút “Delete” 18](#_Toc27054817)

[Hình 2. 14 Chọn từ cần xóa và ấn nút “Save” 18](#_Toc27054818)

[Hình 2. 15 Kiểm tra rỗng 19](#_Toc27054819)

[Hình 2. 16 Xóa ở đầu ngăn 19](#_Toc27054820)

[Hình 2. 17 Xóa ở vị trí tiếp theo 19](#_Toc27054821)

[Hình 2. 18 Ấn vào nút “Edit” 20](#_Toc27054822)

[Hình 2. 19 Nhập thông tin cần sửa và ấn “Save” 21](#_Toc27054823)

# MỞ ĐẦU

## 1. Lý do chọn đề tài

Bảng băm là một cấu trúc dữ liệu lưu trữ một tập hợp cho phép ta có thể nhanh chóng xác định xem một phần tử nào đó có nằm trong tập hợp hay không.Trong một bảng băm có kích thước đủ lớn, thời gian trung bình cho mỗi lần tra cứu là độc lập với số lượng phần tử trong bảng. Nhiều thiết kế bảng băm cũng cho phép chèn thêm và xóa bỏ tùy ý của cặp khóa-giá trị trong thời gian trung bình (hoặc trừ dần) không đổi cho mỗi thao tác.

Trong nhiều trường hợp, bảng băm có hiệu quả hơn so với cây tìm kiếm hoặc bất kỳ cấu trúc dữ liệu tìm kiếm nào. Vì lý do này, chúng được sử dụng rộng rãi trong nhiều loại phần mềm máy tính, đặc biệt là cho mảng kết hợp,lập [chỉ mục cơ sở dữ liệu](https://vi.wikipedia.org/w/index.php?title=Ch%E1%BB%89_m%E1%BB%A5c_c%C6%A1_s%E1%BB%9F_d%E1%BB%AF_li%E1%BB%87u&action=edit&redlink=1), tổ chức [bộ nhớ đệm](https://vi.wikipedia.org/w/index.php?title=B%E1%BB%99_nh%E1%BB%9B_%C4%91%E1%BB%87m&action=edit&redlink=1), và [cấu trúc dữ liệu tập hợp](https://vi.wikipedia.org/w/index.php?title=C%E1%BA%A5u_tr%C3%BAc_d%E1%BB%AF_li%E1%BB%87u_t%E1%BA%ADp_h%E1%BB%A3p&action=edit&redlink=1).

Có thể lập được một chương trình viết bằng ngôn ngữ C shape giúp người học có thể tra cứu được tử Anh-Việt.

## 2. Mục tiêu cần đạt được

Đề tài này được thực hiện nhằm đạt được mục tiêu là hiểu rõ, sâu sắc hơn về phép

băm, các hàm băm. Tìm hiểu ứng dụng quan trọng của nó trong ứng dụng tìm kiếm từ. Thông qua đó nhóm tác giả sẽ xây dựng nên 1 từ điển mà cho phép người dùng tra cứu, thêm từ và xóa từ trong từ điển 1 cách nhanh chóng.

Thiết kế giao diện đẹp và dễ sử dụng.

## 3. Đối tượng và phạm vi nghiên cứu

* Ngôn ngữ C Shape
* Bảng Băm, Hàm Băm
* Danh sách liên kết đơn
* Đọc và ghi file
* Winform C Shape

# CHƯƠNG I:

# TÌM HIỂU VỀ BẢNG BĂM VÀ KỸ THUẬT BĂM

## 1. Bảng băm là gì?

Băm là một kỹ thuật được sử dụng để định danh một đối tượng cụ thể trong một nhóm các đối tượng tương tự. Một số ví dụ về việc sử dụng bảng băm trên thực tế:

Trong trường đại học, mỗi sinh viên được chỉ định một mã sinh viên không giống nhau và qua mã sinh viên đó có thể truy xuất các thông tin của sinh viên đó.

Trong thư viện, mỗi một cuốn sách một mã số riêng và mã số đó có thể được dùng để xác định các thông tin của sách, chẳng hạn như vị trí chính xác của sách trong thư viện hay thể loại của sách đó,…

Trong cả 2 ví dụ trên, các sinh viên và các cuốn sách được “băm” thành các mã số duy nhất(không trùng lặp).

Giả sử rằng bạn có một đối tượng và bạn muốn gán cho nó một cái khóa(key) để giúp tìm kiếm dễ dàng hơn. Để lưu giữ cặp <khóa, giá trị>(nó thường được gọi là <key, value> hơn), bạn có thể sử dụng mảng bình thường để làm việc này; Với chỉ số mảng là khóa và giá trị tại chỉ số đó là giá trị tương ứng của khóa. Tuy nhiên, trong trường hợp phạm vi của khóa lớn và không thể sử dụng chỉ số mảng được, khi đó bạn sẽ cần tới “băm”(hashing).

Trong hashing, các key có giá trị lớn sẽ được đưa về giá trị nhỏ hơn bằng cách sử dụng **hàm băm(hash functions)**. Các giá trị sau đó được lưu trong một cấu trúc dữ liệu gọi là **bảng băm(hash tables)**. Ý tưởng của hashing là đưa các cặp <key, value> về một mảng thống nhất; Mỗi phần tử sẽ được gán một khóa định danh(khóa có được sau khi dùng hàm băm). Bằng việc sử dụng khóa định danh đó, chúng ta có thể truy cập trực tiếp tới nó với độ phức tạp O(1). Thuật toán băm sẽ sử dụng **khóa băm** để tính toán ra khóa định danh của các phần tử hoặc thêm vào bảng băm.

Trong nhiều trường hợp, bảng băm có hiệu quả hơn so với cây tìm kiếm hoặc bất kỳ cấu trúc dữ liệu tìm kiếm nào. Vì lý do này, chúng được sử dụng rộng rãi trong nhiều loại phần mềm máy tính, đặc biệt là cho mảng kết hợp,lập [chỉ mục cơ sở dữ liệu](https://vi.wikipedia.org/w/index.php?title=Ch%E1%BB%89_m%E1%BB%A5c_c%C6%A1_s%E1%BB%9F_d%E1%BB%AF_li%E1%BB%87u&action=edit&redlink=1), tổ chức [bộ nhớ đệm](https://vi.wikipedia.org/w/index.php?title=B%E1%BB%99_nh%E1%BB%9B_%C4%91%E1%BB%87m&action=edit&redlink=1), và cấu trúc dữ liệu tập hợp.

## 2. Lý thuyết về hàm băm

Hàm băm là bất kỳ hàm nào có thể được sử dụng để ánh xạ tập dữ liệu có kích thước tùy ý thành tập dữ liệu có kích thước cố định và đưa vào bảng băm. Các giá trị được trả về bởi hàm băm được gọi là giá trị băm.

Một hàm băm được đánh giá tốt nếu nó đạt được các yêu cầu cơ bản sau:

Dễ tính toán: Nó phải dễ tính toán và bản thân nó không phải là một thuật toán

Phân bố đồng đều: Nó cần phải phân phối đồng đều trên bảng băm, không xảy ra việc tập trung thành các cụm.

Ít va chạm: Va chạm xảy ra khi các cặp phần tử được ánh xạ tới cùng một giá trị băm.

**Chú ý:**Bất kể hàm băm có tốt đên đâu, va chạm vẫn có thể xảy ra. Vì vậy, để duy trì hiệu suất của bảng băm, điều quan trọng là phải quản lý va chạm thông qua các kỹ thuật giải quyết va chạm.

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, ví dụ như, kiểm tra chi-bình phương.

Trong phương pháp [địa chỉ mở](https://vi.wikipedia.org/w/index.php?title=%C4%90%E1%BB%8Ba_ch%E1%BB%89_m%E1%BB%9F&action=edit&redlink=1), 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](https://vi.wikipedia.org/wiki/H%C3%A0m_b%C4%83m_m%E1%BA%ADt_m%C3%A3_h%E1%BB%8Dc) 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.

### Phương pháp phân đoạn

Nếu khoá có kích thước lớn, kích thước thay đổi thì người ta áp dụng phương pháp phân đoạn. Trước hết giá trị khoá phân thành nhiều đoạn khác nhau (có thể trừ đoạn đầu hoặc đoạn cuối) thường mỗi đoạn có độ dài bằng độ dài địa chỉ. Muốn vậy người ta áp dụng các kỹ thuật như:

*a) Tách (spliting):* tách các đoạn ra, xếp mỗi đoạn một hàng, dóng thẳng theo đầu trái hoặc đầu phải.

*b) Gấp (folding):* gấp các đoạn lại theo đường biên tương tự như gấp giấy. Các chữ số rơi vào cùng một chỗ được đặt thành hàng dóng thẳng với nhau.

Sau khi các đoạn đã được tách hoặc gấp chúng sẽ được phối hợp với nhau theo một cách nào đấy. Ví dụ chúng được cộng lại. Từ kết quả thu được lấy một đoạn dài bằng địa chỉ để làm địa chỉ băm hoặc lại áp dụng với nó các kỹ thuật tạo địa chỉ như đã nêu. Giả sử có khoá: 17046329. Bằng phương pháp phân tách ta phân ra các đoạn 3 chữ số kể từ đầu phải rồi cộng lại. Ta có

329

046

017

392

392 được coi là địa chỉ băm ứng với khoá đó.

Còn bằng phương pháp gấp ta sẽ có:

046

923

710

1679

Ta có thể lấy 167 hoặc 679 làm địa chỉ băm.

Với phương pháp này ta cũng thấy các chữ số của khoá đều được tham gia vào việc tạo nên địa chỉ băm tương ứng với nó.

Hàm băm mà nhóm tác giả xây dựng cũng sử dụng phương pháp phân đoạn thông qua việc công tổng tất cả giá trị của chữ theo bảng mã ACII rồi lấy 3 giá trị cuối.

## 3. Giải quyết va chạm

### 3.1 Va chạm là gì?

Hãy đi vào một ví dụ để dễ hình dung nhé. Giả sử rằng bạn cần lưu giữ các string sau đây trong bảng băm: {“abcdef”, “bcdefa”, “cdefab” , “defabc” }

Để tính toán chỉ mục lưu giữ các string này, chúng ta dùng một hàm băm như sau: Chỉ mục của mỗi string sẽ được tính bằng tổng giá trị ASCII của các ký tự trong nó sau đó lấy dư với 599.

Do 5 là số nguyên tố và nó lớn hơn số lượng phần tử, nó có khả năng lập ra các chỉ mục khác nhau(giảm va chạm). Số nguyên tố luôn là lựa chọn tốt nếu bạn muốn dùng phép lấy phần dư. Các giá trị ASCII của a, b, c, d, e và f lần lượt là 97, 98, 99, 100, 101 và 102.

Dễ nhận thấy, các string kia chỉ là các hoán vị của cùng 1 chuỗi, do đó chỉ mục được tạo ra của chúng đều giống nhau và đều bằng 597 % 5 = 2.Lúc này, hàm băm sẽ tính toán và tất cả các string kia đều có cùng 1 chỉ mục. Khi đó, bạn có thể tạo 1 list tại chỉ số đó để lưu các string trong list đó. Như hình ảnh minh họa sau:

Hình 2.1. Va chạm trong bảng băm

Hình 1.1. Va chạm trong bảng băm

Hình 1. 1 Va chạm trong bảng băm



### 3.2. Cách giải quyết va chạm

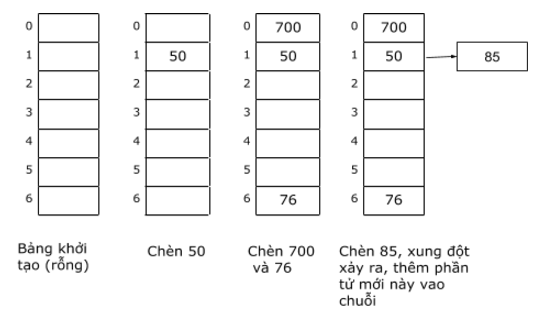
Ta sẽ nghiên cứu ba phương pháp giải quyết xung đột chủ yếu: phân tách chuỗi (separate chaining), băm địa chỉ mở (open addressing) và băm hoàn hảo (perfect hashing).

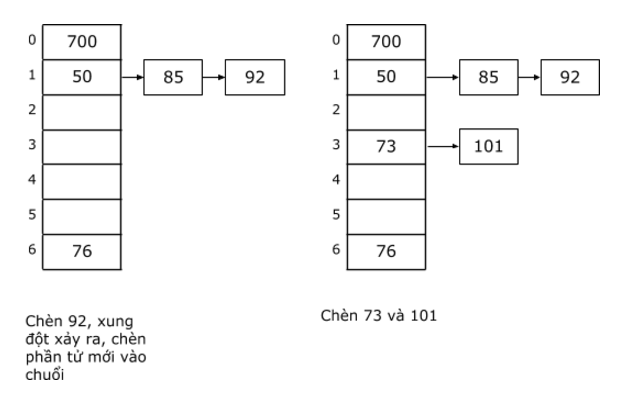
#### ****Kỹ thuật phân tách chuỗi (separate chaining)****

Kĩ thuật phân tách chuỗi là một trong những kĩ thuật phổ biến nhất để giải quyết xung đột trong bảng băm. Kĩ thuật này thường được thực hiện bởi việc sử dụng các danh sách liên kết.

Với kĩ thuật này mỗi phần tử của bảng băm là một danh sách liên kết. Để chèn một phần tử vào bảng băm ta phải chèn nó vào trong một danh sách liên kết. Nếu có xung đột xảy ra như trường hợp hai phần tử có chung giá trị (hash code) thì ta sẽ chèn chúng vào chung một danh sách liên kết.

Giả sử ta có hàm băm chuyển đổi các khóa 50, 700, 76, 85, 92, 73, 101 bằng cách chia cho 7 rồi lấy số dư.





Hình 1. 2 Kỹ thuật phân tách chuỗi

**Ưu điểm:**

Cài đặt đơn giản.

Không phải lo tới kích thước bảng băm, và ta luôn có thể thêm dữ liệu vào bảng bằng cách thêm vào các danh sách liên kết.

Phương thức này thường được sử dụng khi ta không biết tần suất dữ liệu được thêm vào và xóa ra khỏi bảng.

**Nhược điểm:**

Hiệu năng của phương pháp này không tốt bằng phương pháp đánh địa chỉ mở. bởi vì với phương pháp đánh địa chỉ mở thì mọi dữ liệu đều được chứa trong cùng một bảng băm mà không cần trỏ tới một vùng nhớ ngoài bảng.

Đôi khi lãng phí bộ nhớ (như ta nhìn thấy ở ví dụ trên, vị trí 2, 4 và 5 để trống, đôi khi không bao giờ sử dụng tới).

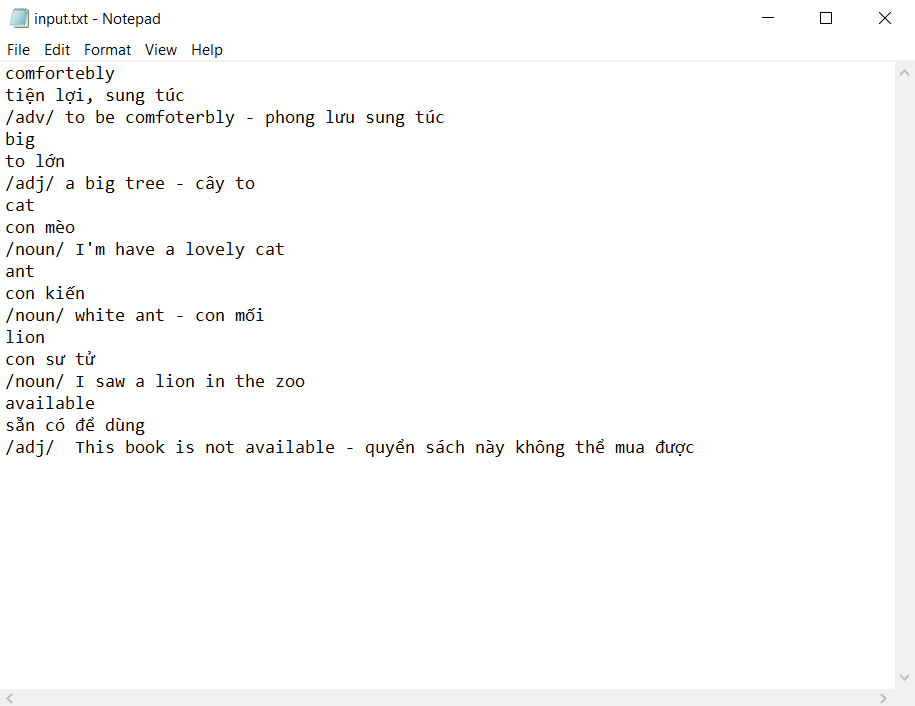
Khi mà chuỗi (danh sách liên kết) trở nên quá dài, lúc đó thời gian cho các thao tác tìm kiếm, xóa phần tử có thể rất tốn thời gian.

Cần thêm bộ nhớ cho các phần tử của danh sách liên kết.

# CHƯƠNG II: BÀI TOÁN ỨNG DỤNG

## 1. Nội dung

### 1.1 File đầu vào



*Hình 2. 1 File txt chứa thông tin đầu vào*

### 1.2 Giới thiệu giao diện



Hình 2. 2 Giao diện chính của từ điển

Tên chương trình: Từ điển Anh – Việt.

Chương trình chứa từ tiếng Anh, nghĩa tiếng Việt và giải thích.

Các chức năng:

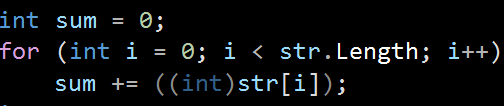
* **English**: Ô chứa từ tiếng Anh
* **Vietnamese**: Ô chứa nghĩa tiếng Việt
* **Ghi chú**: Ô giải nghĩa và nêu ví dụ
* **Danh sách gợi ý**: Ô chứa danh sách từ có trong từ điển
* **Search**: Dịch nghĩa từ tiếng Anh sang tiếng Việt
* **Add**: Thêm từ + nghĩa vào từ điển
* **Delete**: Xóa từ khỏi danh sách có trong từ điển
* **Edit**: Sửa từ
* **Save**: Lưu dùng cho chức năng thêm và xóa từ
* **Exit**: Thoát

## 2. Chương trình minh họa

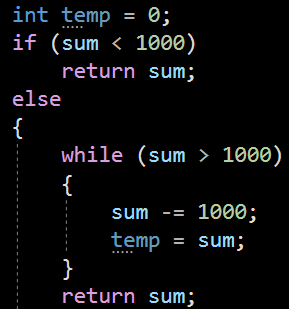
### 2.1. Hàm băm

Hàm băm mà nhóm tác giả xây dựng là chỉ mục của các string sẽ được tính bằng tổng của các mã ASCII của từng ký tự trong string, lí do vì tất cả các từ tiếng Anh kèm theo vị trí sẽ không thể vượt quá 1000 và giảm thiểu được va chạm. VD:

String                                Hash function                               Index  
dog      (1001+1112+1033) 146



Hình 2. 3 Tổng mã ACII

**

Hình 2. 4 Lấy 3 giá trị cuối

### 2.2. Các chức năng của từ điển

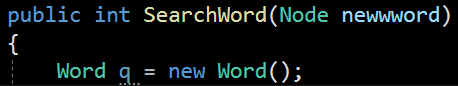
#### a. Tìm kiếm từ

Tư tưởng: Sử dụng hàm băm “băm” từ cần tìm ra thành số i sau đó ta sẽ tìm được ngăn chứa từ ấy là ngăn i. Sau đó kiểm tra danh sách liên kết trong ngăn i xem từ ấy có tồn tại hay không.



Hình 2. 5 Chức năng tìm từ

Tiến hành: Tạo 1 danh sách q và Node mới mang tên newword chứa thông tin từ cần tìm



Hình 2. 6 Khởi tạo danh sách tạm

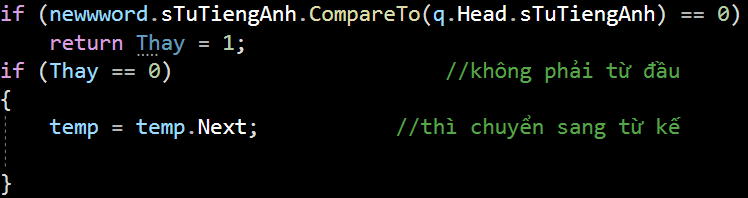
**Lưu ý**: Việc tạo danh sách “thế mạng” q là rất quan trọng nếu không khi thực hiện các thao tác sẽ làm thay đổi giá trị bên trong danh sách gốc dẫn đến lỗi.



Hình 2. 7 “Băm” từ tiếng Anh thành số

Sau đó bước tiếp theo gán q bằng ngăn chứa từ cần tìm đã được “băm” bằng hàm băm trước đó Hashfunction là hàm băm mà nhóm tác giả đã nêu trên.

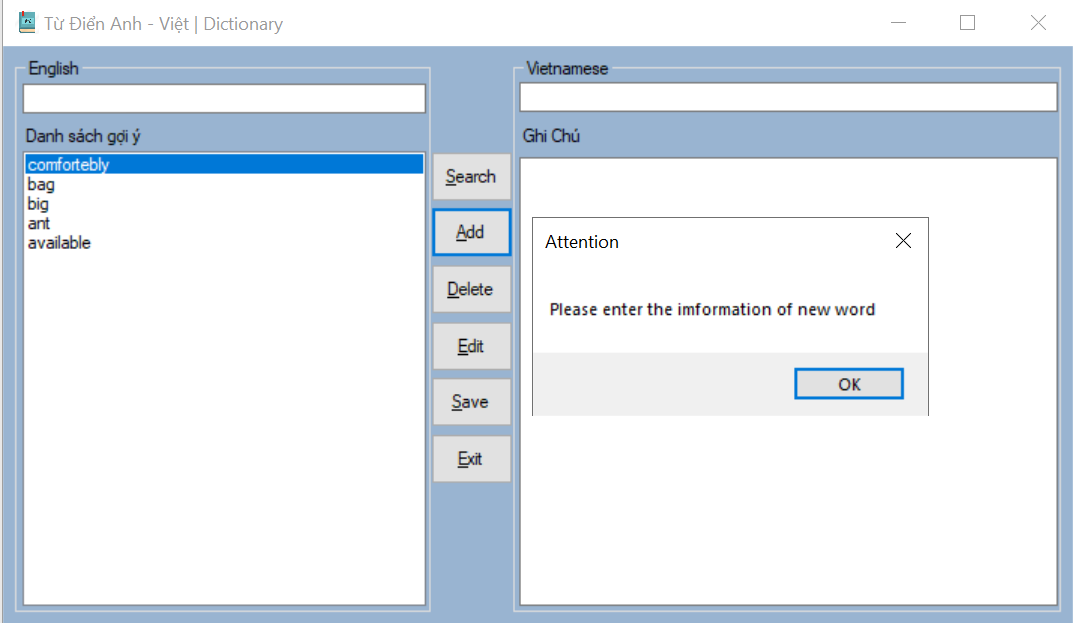
Tiếp theo đến bước kiểm tra chỉ kiểm tra nhưng ngăn nào có giá trị ngăn rỗng bỏ qua. Nếu ngăn không rỗng (chứa ít nhất 1 từ) sử dụng hàm CompareTo so sánh 2 chuỗi (không phân biệt chữ hoa), nếu so sánh từ đầu tiên không phải sau đó qua từ thứ 2 trong ngăn (nếu có) tiếp tục cho đến khi con trỏ trỏ đến “NULL”.



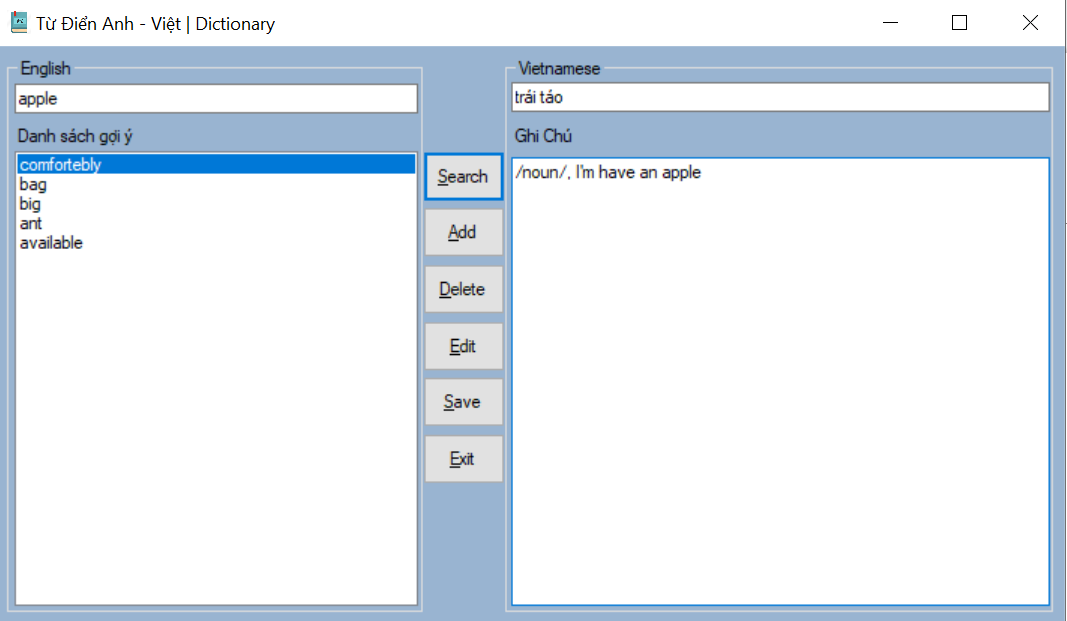
Hình 2. 8 Tiến hành tìm từ

#### b. Thêm từ

Tư tưởng: Tìm ngăn chứa bằng với số i khi tiến hành “băm” từ thành số i. Sau đó tiến hành thêm từ vào danh sách.



Hình 2. 9 Ấn vào nút “Add” để tiến hành thêm từ



Hình 2. 10 Nhập thông tin vào

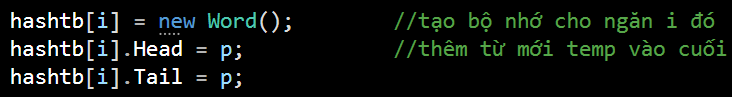
Tiến hành: Đầu tiên vẫn sử dụng hàm băm để “băm” từ cần thêm thành số rồi truy xuất đến ngăn i.



Tiếp đến sử dụng hàm Searchword để kiểm tra sự tồn tại của từ cần thêm.

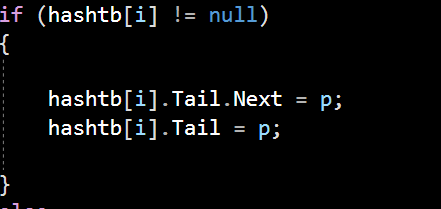


Nếu chưa có thì tiến hành thêm. Chia làm 2 trường hợp: Đầu tiên ngăn rỗng thì tiến hành thêm bình thường.



Hình 2. 11 Thêm vào ngăn rỗng

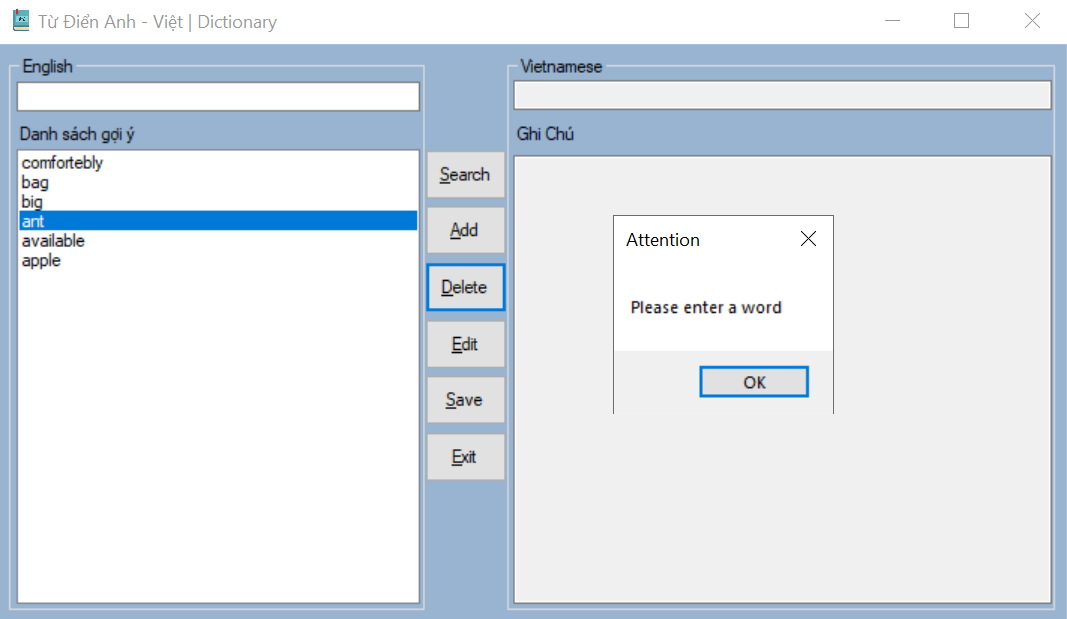
Trường hợp 2 phức tạp hơn là đã tồn tại trước 1 từ trong ngăn thì ta tiến hành thêm vào **SAU** của danh sách.



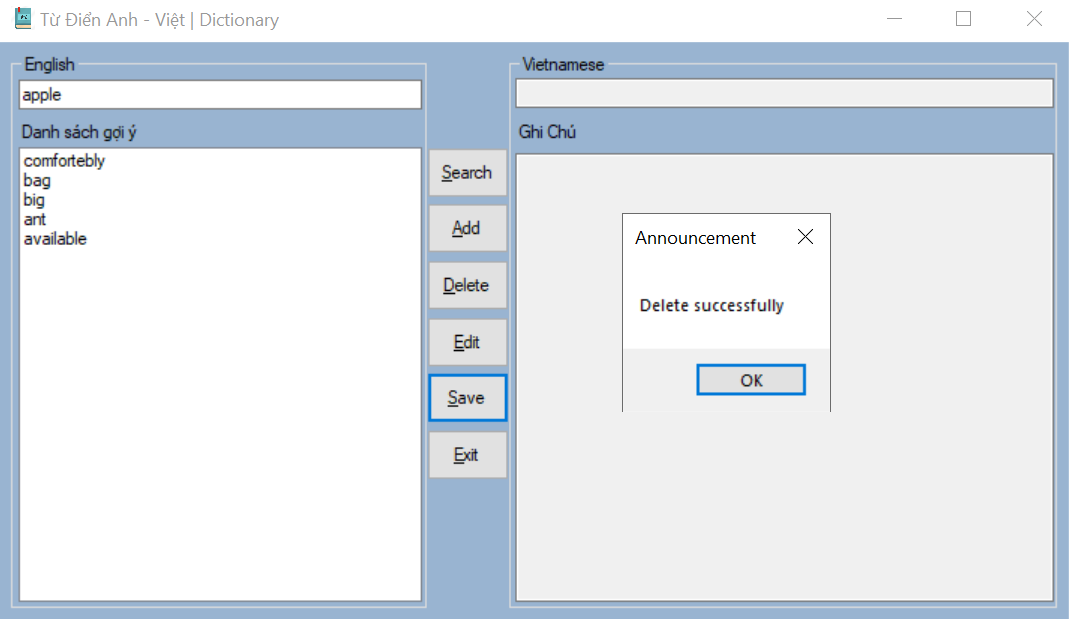
Hình 2. 12 Thêm từ khi ngăn đã tồn tại phần tử

#### c. Xóa từ

Tư tưởng: Tương tự như 2 chức năng tìm kiếm, thêm nhóm tác giả sử dụng hàm băm để “băm” từ cần xóa để tìm ngăn số i rồi tiến hành xóa từ.



Hình 2. 13 Nhấn vào nút “Delete”



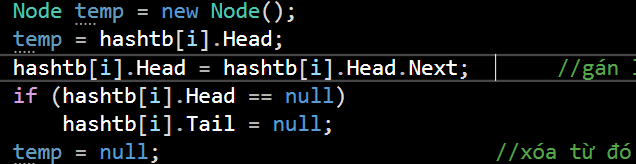
Hình 2. 14 Chọn từ cần xóa và ấn nút “Save”

Tiến hành: Kiểm tra nếu từ cần xóa tồn tại thì tiến hành xóa



Hình 2. 15 Kiểm tra rỗng

Nếu ngay ở đầu danh sách đã tìm thấy từ cần xóa thì t thực hiện như sau:

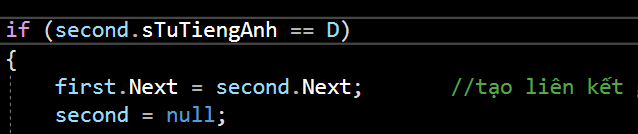


Hình 2. 16 Xóa ở đầu ngăn

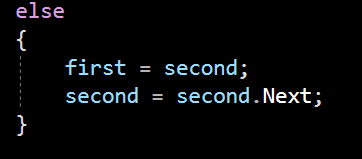
Tạo 1 biến tạm “temp” để tránh ảnh hưởng đến cấu trúc chương trình chính. Gán biến “temp” cho giá trị đầu ngăn. Sau đó gán giá trị đầu cho giá trị kế tiếp rồi tiến hành xóa.

Ngược lại nếu giá trị đầu ngăn không trùng với giá trị cần xóa thì trỏ đến vị trí tiếp theo.



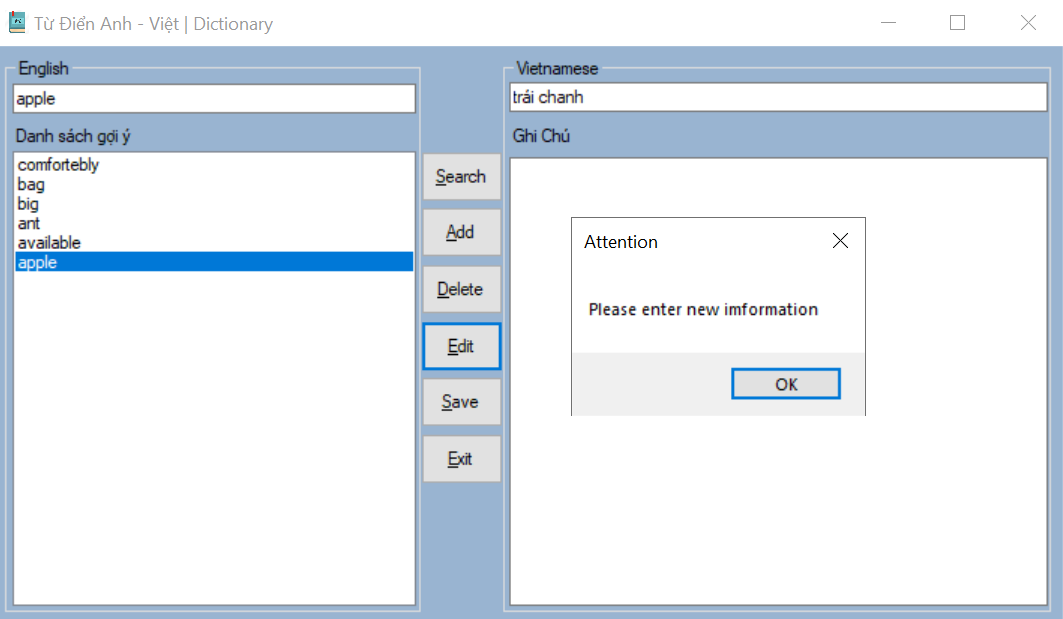


Hình 2. 17 Xóa ở vị trí tiếp theo

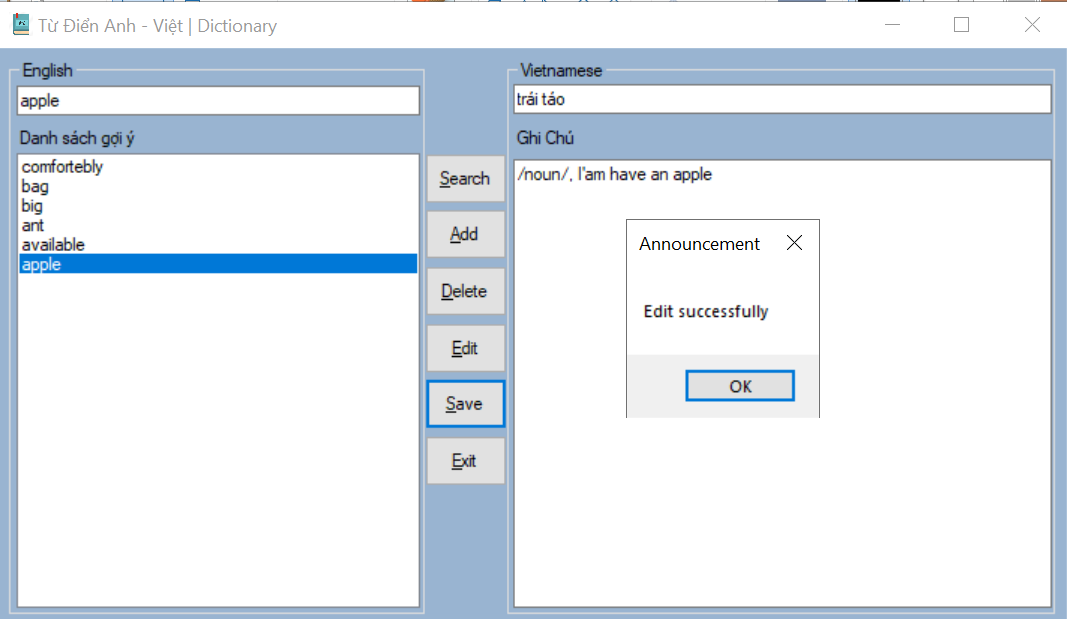


So sánh nếu giá trị thứ 2 trong ngăn trùng với giá trị cần xóa thì tiến hành xóa. Nếu giá trị thứ 2 vẫn không trùng thì trỏ đến vị trí tiếp theo cho đến khi gặp “null” thì dừng lại.

#### d. Sửa



Hình 2. 18 Ấn vào nút “Edit”



Hình 2. 19 Nhập thông tin cần sửa và ấn “Save”

Tìm từ đó có trong từ điển hay không.

Nếu có :

Tiến hành tìm vị trí ngăn của từ đó trong bảng băm.



Tạo 1 Node mới lưu thông tin mới của từ.

Xóa từ cần sửa khỏi bảng băm.



Thêm từ cần sửa (nhưng chứa thông tin mới) vào bảng băm.



#### e. Các hàm khác

Ngoài ra nhóm tác giả còn sử dụng hàm đọc, ghi file và các chức năng trên winform nhưng vì không liên quan đến môn học nên nhóm tác giả xin được phép không đi sâu vào.

# KẾT LUẬN

## 1. Kết quả đạt được

Nhóm tác giả đã đạt được mục đích ban đầu đề ra:

Sử dụng bảng băm xây dựng được từ điển Anh-Việt có các chức năng như tìm kiếm, thêm, xóa và sửa từ.

Xây dựng được giao diện dễ sử dụng đẹp mắt và khả bang thực hiện tương đối nhanh.

Nắm được tương đối lý thuyết bảng băm, hàm băm, phương pháp giải quyết va chạm và ứng dụng được vào bài toán thực tế.

Cũng thông qua đề tài này nhóm tác giả đã nâng cao thêm kỹ năng lập trình bằng ngôn ngữ C#. Học được cách xây dựng giao diện trên winform và khả năng làm việc nhóm.

## 2. Những khó khăn

Trong quá trình thực hiện nhóm tác giả gặp rất nhiều khó khăn.

Đầu tiên là việc tìm hiểu giao diện mới là Window Form đây là một trong những khó khăn lớn nhất trong quá trình thực hiện đồ án. Việc tiếp cận vấn đề mới hoàn toàn làm tăng thời gian thực hiện đồ án lên khá nhiều.

Không có thời gian làm việc cùng nhau do thành viên trong nhóm ở rất xa nhau và quỹ thời gian học khác biệt chỉ có trao đổi online nên dẫn đến bất đồng là điều không thể tránh khỏi.

Cuối cùng là trong quá trình làm việc thì do vấn đề mới nên nhóm tác giả có tham khảo cách thực hiện mỗi người cho một ý kiến nên dẫn đến quyết định đưa ra chưa có chắc chắn ban đầu thực hiện trên C++ nhưng vì giao diện không đẹp nên phải chuyển sang C#. Xem như là làm lại từ đầu.

## 3. Những khuyết điểm

Vì đầy là lần đầu nhóm tác giả thực hiện nên không thể tránh những sai sót các chức năng vẫn chưa được hoàn thiện , hàm băm vẫn chưa được tối ưu hao phí khá nhiều bộ nhớ, giải quyết va chạm giữa các từ trong cùng một ngăn chưa được tốt cũng như giao diện không bát mắt như các ứng dụng trên thị trường nhóm tác giả rất mong nhận được sự góp ý để hoàn thiện sản phầm tốt hơn.

# TÀI LIỆU THAM KHẢO

1. Đỗ Xuân Lôi; *Cấu trúc dữ liệu và giải thuật*; Nhà xuất bản Đại học quốc gia Hà Nội; năm 2006.
2. Nguyễn Thị Tĩnh; *Cấu trúc dữ liệu và giải thuật*; Nhà xuất bản Đại học sư phạm; năm 2000.
3. Lê Văn Vinh; *Giáo trình: Cấu trúc dữ liệu và giải thuật*; NXB Đại học Quốc gia TP. Hồ Chí Minh.
4. <https://nguyenvanhieu.vn/bang-bam-hash-tables/>
5. Bộ 3 video *Cấu trúc dữ liệu nâng cao: Bảng băm*; Ths Nguyễn Tri Tuấn; Đại học Khoa học Tự Nhiên TP.HCM.
6. <https://www.wisdomjobs.com/e-university/data-structures-tutorial-290/open-addressing-7289.html>