**BỘ GIÁO DỤC VÀ ĐÀO TẠO**

**TRƯỜNG ĐẠI HỌC NHA TRANG**

**KHOA CÔNG NGHỆ THÔNG TIN**

**BÁO CÁO THỰC TẬP CƠ SỞ**

**ĐỀ TÀI: MÔ PHỎNG CÁC THUẬT TOÁN SẮP XẾP**

**Giảng viên hướng dẫn : ThS PHẠM THỊ KIM NGOAN**

**Sinh viên thực hiện: TRẦN NHÂN SINH**

**Lớp : 59-CNTT 1**

**MSSV : 59132120**

**Năm học: 2019-2020**

**MỤC LỤC**

[**MỤC LỤC** 2](#_Toc30106292)

[**LỜI CẢM ƠN** 4](#_Toc30106293)

[**CHƯƠNG 1. GIỚI THIỆU ĐỀ TÀI** 5](#_Toc30106294)

[**1.1. Lý do chọn đề tài** 5](#_Toc30106295)

[**1.2. Mục tiêu và nhiệm vụ** 5](#_Toc30106296)

[**CHƯƠNG 2. CƠ SỞ LÝ THUYẾT** 6](#_Toc30106297)

[**2.1. Ngôn ngữ C#** 6](#_Toc30106298)

[**2.1.1. Khái niệm** 6](#_Toc30106299)

[**2.1.2. Ứng dụng của ngôn ngữ lập trình C#** 6](#_Toc30106300)

[**2.2. Thuật toán** 6](#_Toc30106301)

[**2.2.1. Khái niệm thuật toán** 6](#_Toc30106302)

[**2.2.2. Các đặc trưng của thuật toán** 7](#_Toc30106303)

[**2.2.3. Sơ lược các thuật toán sắp xếp hiện nay** 7](#_Toc30106304)

[**2.3. Sơ lược về mô phỏng thuật toán** 8](#_Toc30106305)

[**2.3.1. Khái niệm mô phỏng thuật toán** 8](#_Toc30106306)

[**2.3.2. Quy trình mô phỏng thuật toán** 8](#_Toc30106307)

[**2.3.3. Tác dụng của mô phỏng thuật toán** 9](#_Toc30106308)

[**2.3.4. Kiến trúc của hệ thống mô phỏng thuật toán** 11](#_Toc30106309)

[**2.3.5. Lựa chọn công cụ mô phỏng thuật toán** 12](#_Toc30106310)

[**2.4. Một số yếu cầu đối với mô phỏng thuật toán** 12](#_Toc30106311)

[**2.4.1. Mô tả đúng theo thuật toán** 12](#_Toc30106312)

[**2.4.2. Hệ thống mô phỏng phải được thực hiện theo từng bước** 12](#_Toc30106313)

[**2.4.3. Mô phỏng thuật toán phải có tính động** 13](#_Toc30106314)

[**2.4.4. Phải tạo ra sự phân cấp cho người học** 13](#_Toc30106315)

[**2.4.5. Cấu trúc của mô phỏng thuật toán** 13](#_Toc30106316)

[**CHƯƠNG 3. CHƯƠNG TRÌNH ỨNG DỤNG THUẬT TOÁN SẮP XẾP** 14](#_Toc30106317)

[**3.1. Sơ lược mô phỏng minh họa trên C#** 14](#_Toc30106318)

[**3.1.1. Các chức năng** 14](#_Toc30106319)

[**3.1.2. Các kỹ thuật xử lý** 15](#_Toc30106320)

[**3.1.3. Các kỹ thuật sử dụng** 16](#_Toc30106321)

[**3.2. Các thuật toán sắp xếp đơn giản** 18](#_Toc30106322)

[**3.2.1. Thuật toán Interchange Sort (Đổi chỗ trực tiếp)** 18](#_Toc30106323)

[**3.2.2. Thuật toán Selection Sort (Sắp xếp chọn trực tiếp)** 22](#_Toc30106324)

[**3.2.3. Thuật toán Bubble Sort (Sắp xếp nổi bọt)** 25](#_Toc30106325)

[**3.2.4. Thuật toán Shaker Sort** 28](#_Toc30106326)

[**3.2.5. Thuật toán Insertion Sort (Sắp xếp chèn trực tiếp)** 32](#_Toc30106327)

[**3.2.6. Thuật toán Binary insertion Sort(Chèn nhị phân)** 35](#_Toc30106328)

[**3.2.7. Thuật toán Heap Sort (Sắp xếp vun đống)** 37](#_Toc30106329)

[**3.2.8. Thuật toán Shell Sort** 41](#_Toc30106330)

[**3.3. Các thuật toán sắp xếp sử dụng đệ qui** 44](#_Toc30106331)

[**3.3.1. Thuật toán Quick Sort (Sắp xếp nhanh)** 44](#_Toc30106332)

[**3.3.2. Thuật toán Merge Sort(Sắp xếp trộn)** 48](#_Toc30106333)

[**CHƯƠNG 4. KẾT LUẬN** 52](#_Toc30106334)

[**4.1 Kết quả đạt được** 52](#_Toc30106335)

[**4.2 Hạn chế** 52](#_Toc30106336)

[**4.3 Hướng phát triển** 52](#_Toc30106337)

[**TÀI LIỆU THAM KHẢO** 53](#_Toc30106338)

**LỜI CẢM ƠN**

Lời đầu tiên, em xin chân thành cảm ơn cô Phạm Thị Kim Ngoan đã truyền đạt cho em những kiến thức, kinh nghiệm và giúp đỡ em trong suốt gần một tháng qua trong quá trình thực hiện báo cáo thực tập cơ sở này.

Mặc dù đã cố gắng hoàn thành báo cáo trong phạm vi và khả năng cho phép nhưng chắc chắn không tránh khỏi những thiếu xót. Vì vậy, em rất mong nhận được sự thông cảm, góp ý và tận tình chỉ bảo của thầy cô.

Khánh Hòa, ngày 14 tháng 1 năm 2020

**CHƯƠNG 1. GIỚI THIỆU ĐỀ TÀI**

**1.1. Lý do chọn đề tài**

Như chúng ta đã biết, cấu trúc dữ liệu và giải thuật là một trong những môn học cơ bản của sinh viên ngành Công nghệ thông tin. Các cấu trúc dữ liệu và các giải thuật được xem như là 2 yếu tố quan trọng nhất trong lập trình: Chương trình = Cấu trúc dữ liệu + Giải thuật.

Trong khoa học máy tính, cấu trúc dữ liệu là một cách lưu dữ liệu trong máy tính sao cho nó có thể được sử dụng một cách hiệu quả [1].Giải thuật (hay còn gọi là thuật toán – tiếng Anh là Algorithms) là một tập hợp hữu hạn các chỉ thị để được thực thi theo một thứ tự nào đó để thu được kết quả mong muốn [2].

Để giúp cho sinh viên có thể hình dung, hiểu rõ về Cấu trúc dữ liệu + Giải thuật hơn thì các giảng viên thường sử dụng mô phỏng thuật toán như một công cụ hỗ trợ giúp mô phỏng động bằng đồ họa của một thuật toán và các thay đổi trong cấu trúc dữ liệu của nó trong suốt quá trình thực thi.

Hiện nay, một số hệ thống mô phỏng thuật toán được phát triển, hầu hết các hệ thống phổ biến và tinh vi hơn các hệ thống mà thực tế đang sử dụng.

Thuật toán về sắp xếp rất đa dạng và phong phú. Vì vậy vấn đề “Mô phỏng thuật toán sắp xếp” được chọn để làm báo cáo thực tập cơ sở này.

**1.2. Mục tiêu và nhiệm vụ**

* Nghiên cứu tổng quan về mô phỏng thuật toán.
* Ôn lại, tìm hiểu một số thuật toán thuật toán sắp xếp
* Xây dựng chương trình mô phỏng giúp người học có thể hình dung rõ hơn về quá trình sắp xếp
* Áp dụng kết quả nghiên cứu làm một demo mô phỏng thuật toán sắp xếp trên màn hình console và C#

**CHƯƠNG 2. CƠ SỞ LÝ THUYẾT**

**2.1. Ngôn ngữ C#**

**2.1.1. Khái niệm**

Ngôn ngữ C#(đọc là “C thăng” hay “C sharp”) là một ngôn ngữ lập trình hướng đối tượng được phát triển bởi Microsoft, là phần khởi đầu cho kế hoạch .NET của họ [3].

**2.1.2. Ứng dụng của ngôn ngữ lập trình C#**

* **Ứng dụng trên Windows:** Với sự trợ giúp của bộ khung .Net, “C#” được sử dụng để phát triển các ứng dụng dựa trên các cửa sổ cho máy tính để bàn. Nhiều ứng dụng Windows phổ biến như các công cụ Microsof Office, Skype, Photoshop và Visual Studio được phát triển bằng ngôn ngữ này [8].
* **Các thành phần và điều khiển:** Các thành phần và điều khiển là các thư viện có thể được sử dụng để tạo ra một thứ dễ phân phối và có thể chia sẻ được. Thư viện GPS là một ví dụ tuyệt vời cho một thư viện có thể được một lập trình viên xây dựng và dễ dàng phân phối cho các lập trình viên khác để sử dụng trong các ứng dụng của họ. Nó cũng được sử dụng để xây dựng các thành phần máy chủ và nhiều công việc khác nữa [8].
* **Ứng dụng Web:** Với sự trợ giúp của bộ khung .NET, C# có khả năng tạo ra nhiều ứng dụng web bằng cách sử dụng ASP.NET. Đó là một ngôn ngữ phổ biến khác mà ai ai có thể học ngay lập tức khi muốn làm cho ứng dụng web chạy trơn tru trên một máy chủ web. Các ứng dụng Windows chạy trên cả máy máy chủ cũng như trong trình duyệt của máy khách, tùy thuộc vào cách viết mã. Nếu C# được sử dụng dưới hình thức mã hóa ở backend, thì mã C# chạy trên máy chủ và HTML Frontend chạy trong trình duyệt của máy khách [8].

**2.2. Thuật toán**

**2.2.1. Khái niệm thuật toán**

Thuật toán là một dãy hữu hạn các thao tác được sắp xếp theo một trình tự xác định sao cho sau khi thực hiện dãy các thao tác ấy, từ Input của bài toán ta nhận được Output cần tìm [4].

**2.2.2. Các đặc trưng của thuật toán**

Các thuật toán có một số tính chất chung, đó là:

* *Đầu vào (Input):* Một thuật toán có các giá trị đầu vào từ một tập xác định.
* *Đầu ra (Output):* Từ mỗi tập giá trị đầu vào, thuật toán sẽ tạo ra các giá trị đầu ra. Các giá trị đầu ra chính là nghiệm của bài toán.
* *Tính xác định:* Các bước của thuật toán phải được xác định một cách chính xác.
* *Tính đúng đắn:* Một thuật toán phải cho các giá trị đầu ra đúng đối với mỗi tập giá trị đầu vào.
* *Tính hữu hạn:* Một thuật toán phải tạo ra các giá trị đầu ra sau một số hữu hạn (có thể rất lớn) các bước thực hiện đối với mỗi tập đầu vào.
* *Tính tổng quát:* Thuật toán cần phải áp dụng được cho mọi tập dữ liệu đầu vào của bài toán, chứ không phải chỉ cho một tập đặc biệt các giá trị đầu vào.

**2.2.3. Sơ lược các thuật toán sắp xếp hiện nay**

Trong khoa học máy tính và trong toán học, thuật toán sắp xếp là một thuật toán sắp xếp các phần tử của một danh sách(hoặc một mảng) theo thứ tự tăng (hoặc giảm). Người ta thường xét trường hợp các phần tử cần sắp xếp là các số [9].

Sắp xếp là quá trình xử lý một danh sách các phần tử (hoặc các mẫu tin) để đặt chúng theo một thứ tự thỏa mãn một tiêu chuẩn nào đó dựa trên nội dung thông tin lưu giữ tại mỗi phần tử.

Mức độ hiệu quả của từng giải thuật phụ thuộc vào tính chất của cấu trúc dữ liệu cụ thể mà nó tác động đến.

Có nhiều giải thuật sắp xếp: Selection Sort, Bubble Sort, Insertion Sort, Interchange Sort, Shell Sort, Heap Sort, Quick Sort, Merge Sort, Shaker Sort,….Nhưng trong bài báo cáo này, em xin trình bày và những giải thuật sắp xếp kinh điển để có cái nhìn khoa học hơn về giải thuật sắp xếp.

**2.3. Sơ lược về mô phỏng thuật toán**

**2.3.1. Khái niệm mô phỏng thuật toán**

Mô phỏng thuật toán là quá trình mô tả cấu trúc dữ liệu, thao tác của một chương trình bằng đồ họa [6].Mô phỏng thuật toán được thiết kế để giúp người dùng có thể hiểu thuật toán, đánh giá chương trình và sửa lỗi chương trình.

Một chương trình máy tính chứa các cấu trúc dữ liệu của thuật toán mà nó thực thi. Trong quá trình thực thi chương trình, các giá trị trong cơ sở dữ liệu được thay đổi. Mô phỏng thuật toán sử dụng biểu diễn đồ họa để biểu diễn cấu trúc dữ liệu và chỉ ra sự thay đổi giá trị trong cơ sở dữ liệu trong mỗi trạng thái. Thông qua đó, người sử dụng có thể xem được từng bước thực thi chương trình và nhờ vậy có thể hiểu chi tiết được thuật toán.

Mô phỏng thuật toán cũng được dùng để đánh giá một chương trình đã có bằng cách cung cấp các mô phỏng cho các thành phần của hệ thống, nhờ đó có thể kiểm tra được hiệu năng của hệ thống.

Bên cạnh việc giúp người sử dụng hiểu hơn về hệ thống, mô phỏng thuật toán còn được dung để giúp thực hiện quá trình dò lỗi dễ dàng hơn. Để sử dụng mô phỏng thuật toán trong quá trình dò lỗi của một chương trình, người sử dụng chú thích vào các trạng thái của chương trình để tạo ra các lệnh mô phỏng, sau đó chúng sẽ được đưa vào hệ thống mô phỏng thuật toán để tạo mô phỏng. Người sử dụng có thể xem chương trình của họ đã thực hiện như thế nào, các giá trị dữ liệu ở mỗi bước và một bước sẽ ảnh hưởng tới các bước sau như thế nào. Nó sẽ giúp người sử dụng tìm ra tất cả các lỗi có thể xảy ra trong chương trình.

**2.3.2. Quy trình mô phỏng thuật toán**

**Bước 1: Phân tích thuật toán**

Để thực hiện quá trình mô phỏng thì việc đầu tiên là phải phân tích kỹ thuật toán đó. Một thuật toán cần dùng đến chương trình mô phỏng thì thường thuật toán không đơn giản. Xác định dữ liệu vào và dữ liệu ra của thuật toán, tức là xác định thuật toán xây dựng trên cấu trúc dữ liệu nào, xác định các trường hợp cần xét của dữ liệu, giá trị biến để thoát khỏi vòng lặp…

Chẳng hạn, cho dãy số nguyên, yêu cầu tìm phần tử lớn nhất. Với bài toán này, xác định dữ liệu vào là mảng một chiều, với dữ liệu kiểu mảng xác định cách nhập mảng… hay bài toán danh sách cán bộ với hai thao tác thường xuyên thực hiện là bổ sung và loại bỏ thì dùng cấu trúc dữ liệu mảng không còn phù hợp mà nên dùng kiểu danh sách nối đơn, từ đó xác định thuật toán tác động trên danh sách móc nối.

**Bước 2: Xây dựng mô hình mô phỏng dữ liệu vào và dữ liệu ra**

Mô phỏng dữ liệu vào là cách chọn hình thức hiện thị cho cấu trúc dữ liệu tương ứng với giải thuật. Việc lựa chọn hình thức mô phỏng cho dữ liệu vào quyết định tính hiệu quả của chương trình mô phỏng.

**Bước 3: Tách thuật toán thành nhiều bước nhỏ**

Việc tách thuật toán thành nhiều bước nhỏ tương tự như phân tích bài toán theo hướng modul hóa, giúp quá trình mô phỏng dễ dàng và chi tiết hơn.

**Bước 4: Tổng hợp các các bước mô phỏng thành thuật toán hoàn chỉnh**

**Bước 5:  Kiểm nghiệm giải thuật thông qua dữ liệu ra của từng bước nhỏ**

**2.3.3. Tác dụng của mô phỏng thuật toán**

Các hệ thống mô phỏng thuật toán được sử dụng rộng rãi như công cụ hỗ trợ giảng dạy trong ngành giáo dục khoa học máy tính. Một số nghiên cứu thực nghiệm đã ước lượng hiệu quả của chúng trong giáo dục và kết quả nhận được có thay đổi. Cụ thể là:

Brown (1984) đã sử dụng BALSA\_I để dạy một khóa giới thiệu lập trình và một khóa “cấu trúc dữ liệu và giải thuật”. Hệ thống được sử dụng như một chương trình trực quan trong khóa giới thiệu, và như một người mô phỏng thuật toán mức cao trong lớp cấu trúc dữ liệu. Ông ta báo cáo rằng việc sử dụng các hoạt cảnh mô phỏng để phụ thêm vào thuyết trình dẫn tới “những lợi ích có thể chứng minh được trong việc tăng tốc độ hiểu biết” qua thuyết trình truyề thống. Stasko (1997) đx sử dụng Samba, chương trình mô phỏng của hệ thống XTango dạy một khóa thuật toán khoa học máy tính. Những sinh viên được yêu cầu sử dụng hệ thống có thêm vào mô phỏng cho các chương trình ấn định cảu họ. Các kết quả thua được cho biết rằng những sinh viên thích các mô phỏng và những mô phỏng đó có thể làm tăng tính sang tạo của các sinh viên. Hơn nữa, sự hiểu biết của sinh viên về thuật toán được tăng lên nhờ việc mô phỏng [7].

Tuy nhiên, sử dụng thuật toán trong việc dạy học không phải lúc nào cũng thành công. Các nhà giáo dục đã làm các thực nghiệm và thu được các kết quả pha trộn. Stasko et al (1993) đã chỉ ra một thí nghiệm bằng việc dạy hai nhóm sinh viên với hai cách thuyết trình khác nhau. Cả hai nhóm sinh viên này cùng nghiên cứu thuật toán “Pairing heap” (ghép đôi đống). Một nhóm học thuật toán dựa vào sự mô tả văn bản và nhóm kia cũng nhận các tài liệu đó nhưng có thêm sự trợ giúp bằng các chương trình mô phỏng thuật tóa. Mặc dầu những kết quả chỉ ra rằng nhóm thứ hai đạt được nhiều điểm hơn nhóm kia, nhưng không có điểm nổi trội nào có thể được kết luận là nhờ sự trợ giúp của mô phỏng.

Tương tự, Byrne et al (1996) đã chủ đạo hai thí nghiệm mà trong đó các kết quả chỉ ra rằng lợi ích của mô phỏng không phải là hiển nhiên. Những kết quả pha trộn này đã gây ra chán nản, nhưng không có điểm nổi trội nào có thể được kết luận là nhờ sự trợ giúp của mô phỏng.

Những kết quả này cho thấy rằng để mô phỏng thuật toán có hiệu quả và có lợi cho người dung, thì việc thiết kế cho thích hợp và cách thức mô phỏng là những yếu tố quan trọng. Để mô phỏng thuật toán có hiệu quả thì hệ thống mô phỏng cần phải đáp ứng những điều sau:

* Truy cập mở (Open access): Người dùng có thể truy cập hệ thống mô phỏng mở. Hơn nữa, nếu có cài đặt hệ thống mô phỏng trong trường học, thì họ có thể truy cập tới hệ thống này từ nhà hoặc từ bất cứ nới nào khác.
* Mô phỏng một cách có điều khiển (Control animation): Người dùng có thể tự tạo tập dữ liệu của chính mình khi sử dụng hệ thống mô phỏng. Trong khi các tập dữ liệu được cài đặt sẵn cũng có thể giúp đỡ sinh viên có những sự hiểu biết ban đầu, hệ thống nên có cả 2 tùy chọn này.
* Tương tác (Ineractivity): Hệ thống mô phỏng phải cung cấp được sự tương tác giữa người dung và hệ thống. Sự tương tác bao gồm: người dung xem theo từng bước, hủy, chạy nhanh tới một bước mon muốn, hay xem lại từ đầu,…
* Lịch sử (History): Hệ thống mô phỏng cho phép người dung xem lại các bước trước trong quá trình thực hiện
* Phản hồi (Feedback): Phải tiếp thu phản hồi của sinh viên về việc sử dụng hệ thống mô phỏng để ước lượn hiệu quả của hệ thống cũng như để cải thiện hệ thống.

**2.3.4. Kiến trúc của hệ thống mô phỏng thuật toán**

Đa số các hệ thống mô phỏng thuật toán có những thư viện hỗ trợ thủ tục mô phỏng và giao diện mô phỏng. Vài hệ thống mô phỏng đòi hỏi phải đưa vào trực tiếp bằng tay những thông điệp gửi tới các thủ tục mô phỏng trong chương trình thực hiện thuật toán. Những hệ thống mô phỏng thuật toán ra đời sớm như: BALSA và TAGO là sự kiện – điều khiển, nghĩa là chúng có một chương trình phát sinh những sự kiện trong dạng những thông đệp tới một máy chủ thông điệp. Máy chủ thông đệp chuyển thông điệp tới những cảnh quan tương ứng. Một cảnh quan là một cửa sổ trong một thiết bị màn hình nơi người dung nhìn những đối tượng mô phỏng. Thông điệp bao gôm thông tin của một đối tượng mô phỏng. Sau khi cảnh quan nhận thông điệp, nó tính toán lại đối tượng và kéo lại nó trên cảnh quan [5].

Vài hệ thống gần đây được viết bằng Java và tất cả đều có những kiến trúc tương tự nhau. Những hệ thông này chấp nhận framework của TANGO như kiến trúc của nó. Tất cả các hệ thống sẽ gồm có 3 thành phần, các hàm mô phỏng, kênh mô phỏng và trình diễn mô phỏng:

* Các hàm mô phỏng: Chứa các thư viện để vẽ các đối tượng mô phỏng trên thiết bị màn hình.
* Màn hình trình diễn mô phỏng: Cung cấp một môi trường đồ họa để trình diễn mô phỏng trên thiết bị màn hình tới người dung cuối
* Kênh mô phỏng dịch các lệnh kịch bản thành các lệnh mô phỏng tương ứng và chuyển qua những tham số điều khiển của đối tượng mô phỏng tới các hàm mô phỏng.
* Các hàm mô phỏng vẽ đối tượng được mô phỏng theo các tham số điều khiển của đối tượng đó tới Animation viewer.
* Các tham số điều khiển bao gồm tọa độ x và y chỉ rõ nơi đối tượng được mô phỏng xuất hiện trong Animation viewer hoặc mày sắc của đối tượng được mô phỏng.

**2.3.5. Lựa chọn công cụ mô phỏng thuật toán**

Trong mục này, chúng ta sẽ phân tích cách tiếp cận khác để xây dựng hệ thống mô phỏng và tính khả thi của chúng. Chúng ta cũng sẽ ước lượng một vài công cụ mô phỏng thuật toán thích hợp để xây dựng hệ thống mô phỏng thuật toán. Công cụ thích hợp nhất sẽ được lựa chọn và các căn chỉnh trên sự lựa chọn này sẽ được cung cấp.

Có ba cách tiếp cận có thể để xây dựng hệ thống mô phỏng phân tác. Cách tiếp cận đầu tiên sẽ xây dựng hệ thống từ đầu nhờ việc sử dụng ngôn ngữ C#. Cách tiếp cận thứ hai sẽ lựa chọn hệ thống mô phỏng thuật toán có mục đích chung thích hợp để xây dựng các thành phần tương tác của hệ thống phân tách từ đầu. Cách tiếp cận cuối cùng là lựa chọn một hệ thống mô phỏng thuật toán phân tách đã tồn tại và sửa đổi hệ thống đó thành hệ thống cuối cùng.

**2.4. Một số yếu cầu đối với mô phỏng thuật toán**

**2.4.1. Mô tả đúng theo thuật toán**

Thuật toán được đưa ra mô phỏng phải chính xác, các bước thực hiện thuật toán phải trực quan và phản ánh đúng theo nội dung thuật toán đã đưa ra để đảm bảo tính đúng đắn của thuật toán.

Để kiểm tra tính đúng đắn của thuật toán, ta có thể cài đặt giải thuật đó trên máy tính rồi đưa vào các bộ dữ liệu xác định, lấy kết quả thu được xác định với kết quả đã biết. Bộ dữ liệu đưa vào phải đảm bảo kết quả thu được phải vét kín các trường hợp nghiệm của bài toán (trường hơp jthoong thường và các trường hợp đặc biệt). Làm theo cách này thì không chắc chắn, ta chỉ phát hiện được thuật toán sai chứ không khẳng định được luôn đúng. Tính đúng đắn chỉ có thể khẳng định bằng phương pháp chứng minh toán học

**2.4.2. Hệ thống mô phỏng phải được thực hiện theo từng bước**

Thuật toán thường là trừu tượng, nếu để chương trình chạy tự động thì người dung sẽ khó hiều. Vì vậy, cần phải có chế độ thực hiện mô phỏng thuật toán theo từng bước, để người học có thể quan sát, theo dõi sự thay đổi giá trị của từng biến. Nhờ đó, sẽ giúp cho người học hiểu thuật toán rõ hơn và nhanh hơn.

**2.4.3. Mô phỏng thuật toán phải có tính động**

Để mô tả trực quan hóa quá trình thực hiện của thuật toán ta nên đưa vào hình ảnh động (có thể có âm thanh) để thể hiện trong mọi trường hợp để đảm bảo thời gian thực thi tốt nhất

Một thuật toán được mô phỏng phải đảm bảo là thuật toán tốt, dễ hiểu và đúng đắn. Muốn vậy ta phải thử nghiệm trong các trường hợp dữ liệu ngẫu nhiên, tốt nhất, xấu nhất. Nếu thuật toán vẫn chạy tốt và trong một thời gian cho phép thì thuật toán mới hiệu quả. Ta không thể chấp nhận một thuật toán đúng mà thời gian chạy quá lớn.

**2.4.4. Phải tạo ra sự phân cấp cho người học**

Đối tượng học thuật toán thường là các sinh viên. Họ có trình độ tiếp thu khác nhau, nên ta phải đưa ra nhiều chế độ thao tác khác nhau để người học được phép lựa chọn.

**2.4.5. Cấu trúc của mô phỏng thuật toán**

**INPUT**   **ALGORITHM**  **OUTPUT**

-Dữ liệu mẫu

-Dữ liệu trực tiếp

Cấu trúc dữ liệu trừu tượng

-Tự động

-Từng bước

Độ phức tạp của thuật toán

**CHƯƠNG 3. CHƯƠNG TRÌNH ỨNG DỤNG THUẬT TOÁN SẮP XẾP**

**3.1. Sơ lược mô phỏng minh họa trên C#**

**3.1.1. Các chức năng**

Ở phần mô phỏng minh họa trên C# thì phần mềm của em có các chức năng sau:

+ Bao gồm 10 thuật toán sắp xếp tiêu biểu trong cấu trúc dữ liệu và giải thuật.

+ Tạo mới ngẫu nhiên một dãy số các phần tử theo từng thuật toán sắp xếp tương ứng.

+ Xóa mảng: Khi thực hiện xóa mảng tất cả các phần tử trên màn hình chính sẽ không còn xuất hiện nữa.

+ Hủy quá trình: Khi thực hiện hủy quá trình sắp xếp các phần tử trên màn hình chính trở về trạng thái ban đầu chưa sắp xếp.

+ Giới hạn số phần tử được phép thêm vào dãy số (tối đa là 15 phần tử).

+ Hỗ trợ chức năng tạm dừng hay tiếp tục chạy ứng dụng.

+ Điều chỉnh tốc độ thực thi các hoạt động của các thuật toán sắp xếp trong lúc chạy.

+ Có cửa sổ hiển thị code C++ của các thuật toán sắp xếp tương ứng nhằm đối chiếu với hình ảnh mô phỏng.

+ Hiển thị ý tưởng thuật toán và hiện trang dãy số thay đổi khi sắp xếp.

+ Khởi tạo dãy số cần sắp xếp bằng cách click vào phần khởi tạo rồi chọn chế độ.

+ Tạo ngẫu nhiên với số phần tử ngẫu nhiên.

+ Nhập bằng tay: nhập từng phần tử với số phần tử tương ứng khi nhập vào ô số phần tử.

+ Lựa chọn chế độ debug cho phép chạy từng dòng lệnh theo mã code C++ trên Form chính.

+ Mảng chưa sắp xếp: Hiển thị mảng trước khi sắp xếp.

+ Hiển thị quá trình sắp xếp và kết quả sau khi sắp xếp xong của dãy số.

+ Hiển thị thời gian sắp xếp

+ Cho đổi tông màu hiển thị của phần mềm

**3.1.2. Các kỹ thuật xử lý**

**3.1.2.1. Phân tích hiệu ứng**

Để mô phỏng một phần tử trên mảng ta cần tạo một user controls gọi là Node để lưu giá trị của phần tử mảng và hiển thị trực tiếp lên GUI, có thanh trackbar cho người dùng có thể dễ dàng thay đổi tốc độ của quá trình sắp xếp

Trong lúc các phần tử di chuyển thì phần code thể hiện của thuật toán cũng được chọn từng dòng tương ứng với việc sắp xếp các phần tử.

Việc hiển thị quá trình sắp xếp cũng cho người dùng dễ theo dõi và dễ hiểu thuật toán hơn.

Ngoài ra các biến trong thuật toán còn được biễu diễn trên form một cách trực quan sao cho phù hợp với vị trí của phần tử và có giá trị tương ứng với giá trị của biến trong thuật toán.

**3.1.2.2. Phân tích chức năng**

* **Khởi tạo mảng (3 cách) :**
* Tạo ngẫu nhiên : tạo một mảng với các phần tử có giá trị bất kì và số phần tử bất kì
* Nhập bằng tay : cho người dùng nhập giá trị của từng phần tử một. Người dùng có thể thay đổi giá trị của một phần tử bất kì theo ý muốn.
* Nhập một dãy : cho người dùng nhập một dãy số liên tiếp và tạo mảng các phần tử dựa vào dãy số vừa nhập. Người dùng có thể lưu lại dãy vừa nhập cho các lần nhập sau nếu muốn.
* **Bảng điều khiển:**
* Bắt đầu sắp xếp : bắt đầu sắp xếp các phần tử dựa vào thuật toán được chọn
* Tạm dừng : cho tạm dừng quá trình sắp xếp
* Thời gian thực hiện : cho xem tổng thời gian thực hiện quá trình sắp xếp
* **Các chức năng phụ:**
* Thay đổi tốc độ quá trình sắp xếp
* Đa ngôn ngữ hổ trợ Anh-Việt
* Cập nhật thông tin cho quá trình sắp xếp
* Cho phép chạy thuật toán ở hai hướng là tăng và giảm
* Thông qua mục cài đặt ta có thể thay đổi các thuộc tính của phần mềm như : font chữ của các khung hiển thị, màu, kích cỡ của các phần tử, khoản cách giữa các phần tử

**3.1.3. Các kỹ thuật sử dụng**

**3.1.3.1. Xử lý bất đồng bộ với Thread và Task**

Để có thể mô phỏng việc di chuyển của các phần tử khi sắp xếp ta cần sử dụng kỹ thuật lập trình bất đồng bộ. Để lập trình bất đồng bộ ta có thể sử dụng Thread hoặc Task hoặc sử dụng cả hai.

Để có thể nhìn thấy các phần tử di chuyển mượt mà khi sắp xếp ta cần dùng đến hàm Thread.Sleep(<thời gian>) để cho tiến trình dừng lại trong một khoản thời gian nhất định và giúp cho người dùng kịp nhận ra sự thay đổi vị trí của các phần tử.

Để chọn từng dòng code trong khung Code C/C++ thì ta cần tạo một tiến trình chạy song song với tiến trình chính để khi tiến trình thực hiện sẽ không ngăn cản (block) tiến trình chính tiếp nhận các sự kiện (event) xảy ra.

Để có thể có hai chuyển động (hai hàm di chuyển phần tử) cùng chạy một lúc thì ta cần tạo ra hai tiến trình chạy song song với tiến trình chính.

Khi sử dụng Thread ta có thể hủy Thread bất kì khi nào mình muốn nhưng Thread lại không hỗ trợ các thư việc giúp thông báo khi tiến trình đã hoàn tất (callback). Vì thế ta sử dụng Task, Task hỗ trợ đủ các hàm thông báo việc tiến trình kết thúc nhưng lại thiếu hàm hủy Task đang thực hiện, việc hủy Task phải thông qua một CancellationTokenSource và cần phải thực hiện nhiều việc mới có thể có thể hủy một Task đang thực hiện.

Vì các nhược điểm và ưu điểm trên nên ta kết hợp việc sử dụng Thread và Task lại để thực hiện quá trình sắp xếp. Ta cho tạo một Thread để chạy việc chọn từng dòng code, bên trong Thread đó ta tạo ra hai Task con đê thực hiện việc di chuyển các phần tử. Vì vậy ta vừa có thể báo việc hoàn tất di chuyển của các phần tử và vừa có thể hủy quá trình bất cứ khi nào.

Khi xử lý đa tiến trình ta cũng gặp phải một vấn đề khá quan trọng đó là việc các tiến trình không cho phép tiến trình khác thay đổi hay chỉnh sửa các control được tạo ra bên trong tiến trình của mình. Ví dụ: khi ta tạo ra tiến trình con để thay đổi đoạn Text trên một Button của tiến trình chính thì Compiler sẽ báo lỗi. Khi đó ta cần thông báo cho Compiler biết là sẽ bỏ qua và không kiểm ta lỗi này

Việc tạm dừng các tiến trình ta sẽ sử dụng đên ManualResetEvent đây là một lớp có thể dùng để tạm dừng (block) các tiến trình đang chạy và tiếp tục (resume) các tiến trình khi cần.

**3.1.3.2. Xử lý việc thay đổi tốc độ sắp xếp**

Do ở mỗi vòng lặp khi thay đổi tọa độ của phần tử ta đều lấy giá trị tốc độ hiện tại được lưu trong tham số nên khi thay đổi giá trị tốc độ hiện tại thì các vòng lặp trong tương lai sẽ sử dụng giá trị tốc độ mới. Điều này sẽ làm thay đổi tốc độ di chuyển các phần tử, dẫn đến thay đổi tốc độ sắp xếp

**3.1.3.3.** **Xử lý việc di chuyển các phần tử và các biến**

Để xác định được tọa độ đặt và tọa độ di chuyển các phần tử và các biến ta cần dựa vào chiều dài, rộng của panel xác định được tọa độ gốc. Từ tọa độ gốc ta dựa vào vị trí phần tử trong mảng để tính ra tọa độ mà phần tử sẽ được đặt trên panel và cũng dùng cách tương tự để xác định tọa độ đích khi cần di chuyển phần tử.

Đối với các biến có trong các thuật toán sắp xếp ta cũng làm tương tự như cách trên.

Ngoài ra các biến và các phần tử cũng được đổi màu để người sử dụng dễ dàng phân biệt.

**3.1.3.4.** **Xử lý việc tính thời gian thực hiện quá trình sắp xếp**

Để xác định được tọa độ đặt và tọa độ di chuyển các phần tử và các biến ta cần dựa vào chiều dài, rộng của panel xác định được tọa độ gốc. Từ tọa độ gốc ta dựa vào vị trí phần tử trong mảng để tính ra tọa độ mà phần tử sẽ được đặt trên panel và cũng dùng cách tương tự để xác định tọa độ đích khi cần di chuyển phần tử.

Đối với các biến có trong các thuật toán sắp xếp ta cũng làm tương tự như cách trên.

Ngoài ra các biến và các phần tử cũng được đổi màu để người sử dụng dễ dàng phân biệt.

**3.1.3.5.** **Xử lý việc lưu thông tin cài đặt của người dùng**

Các thông tin mà người dùng chỉnh sửa trong phần cài đặt sẽ được lưu vào mục Cài Đặt của phần mềm bao gồm các thông tin như: màu phần tử đang xét, màu phần tử đã xét, màu nền của phần tử, số phần tử mặc định, kích cỡ phần tử, khoảng cách giữa các phần tử.

**3.1.3.6.** **Xử lý debug**

Để xứ lý debug chạy từng dòng dùng manualresetevent set đợi trong một khoảng thời gian vô tận khi được sign.

**3.2. Các thuật toán sắp xếp đơn giản**

**3.2.1. Thuật toán Interchange Sort (Đổi chỗ trực tiếp)**

**3.2.1.1. Nghịch thế**

Một cặp giá trị (a, b) được gọi là nghịch thế khi a và b không thỏa điều kiện sắp thứ tự

**Ví dụ:** Cho mảng a gồm 6 phần tử: **14 29 -1 10 5 23**

**Yêu cầu:** Liệt kê các **cặp giá trị nghịch thế** trong mảng sau, biết rằng yêu cầu là sắp xếp mảng **tăng dần**

**Kết quả:**

**Duyệt lần 1: (14, -1) – (14, 10) – (14, 5)**

**Duyệt lần 2: (29, -1) – (29, 10) – (29, 5) – (29, 23)**

**Duyệt lần 3: Không có cặp nghịch thế nào**

**Duyệt lần 4 : (10, 5)**

**Duyệt lần 5 và 6: Không có cặp nghịch thế nào**

**3.2.1.2. Ý tưởng giải thuật**

Giải thuật sắp xếp bằng phương pháp Interchange Sort (Đổi chổ trực tiếp) được thực hiện bằng cách duyệt qua tất cả các cặp giá trị trong mảng và nếu gặp **cặp nghịch thế** sẽ thực hiện việc **hoán vị** (đổi chỗ) cho nhau.

**3.2.1.3. Cài đặt mã giả**

* **Bước 1**: i = 0; //bắt đầu từ đầu dãy
* **Bước 2**: j = i + 1; //tìm các nghịch thế với a[i]
* **Bước 3**:

Trong khi j < N thực hiện

Nếu a[j] < a[i]//Xét cặp a[i], a[j]

Swap(a[i], a[j]);

j = j + 1;

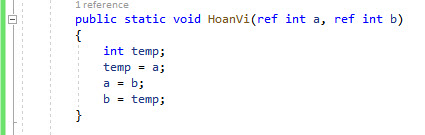
* **Bước 4**: i = i + 1;

Nếu i < N – 1: Lặp lại Bước 2.

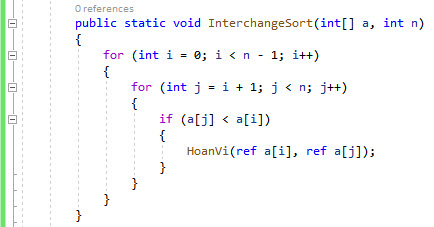
Ngược lại: Dừng

**3.2.1.4. Mô phỏng trên Cửa sổ console**

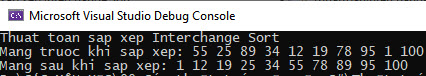
* Hàm hoán vị:



* Hàm sắp xếp Interchange Sort

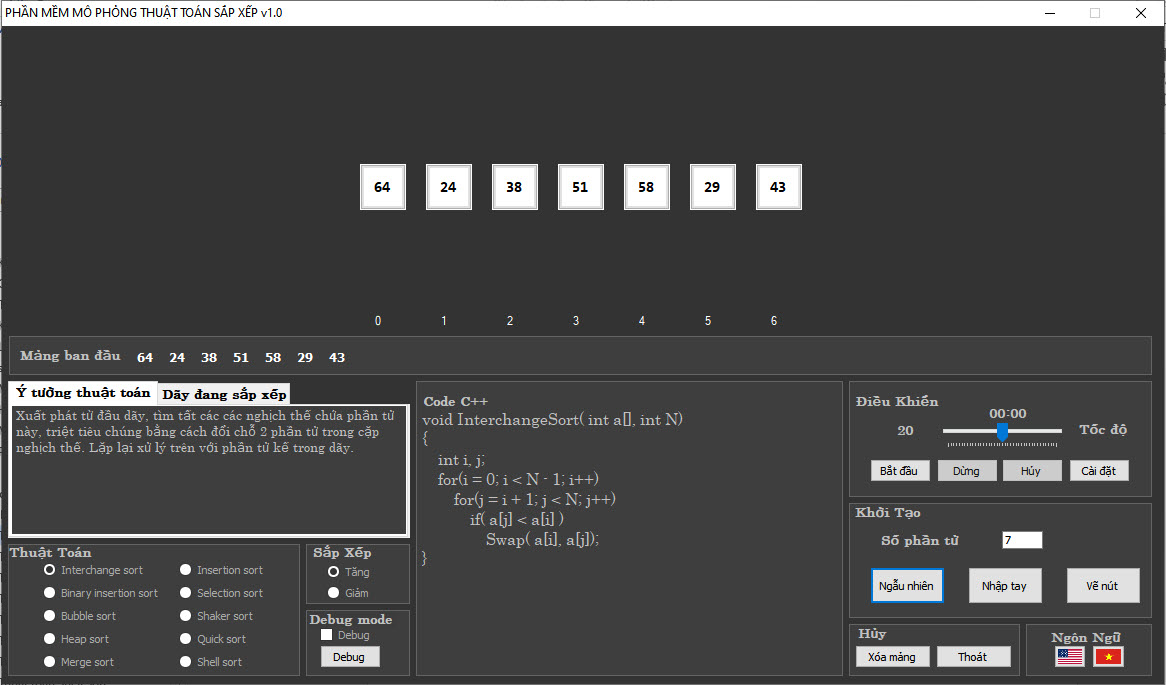


* Kết quả trên màn hình console

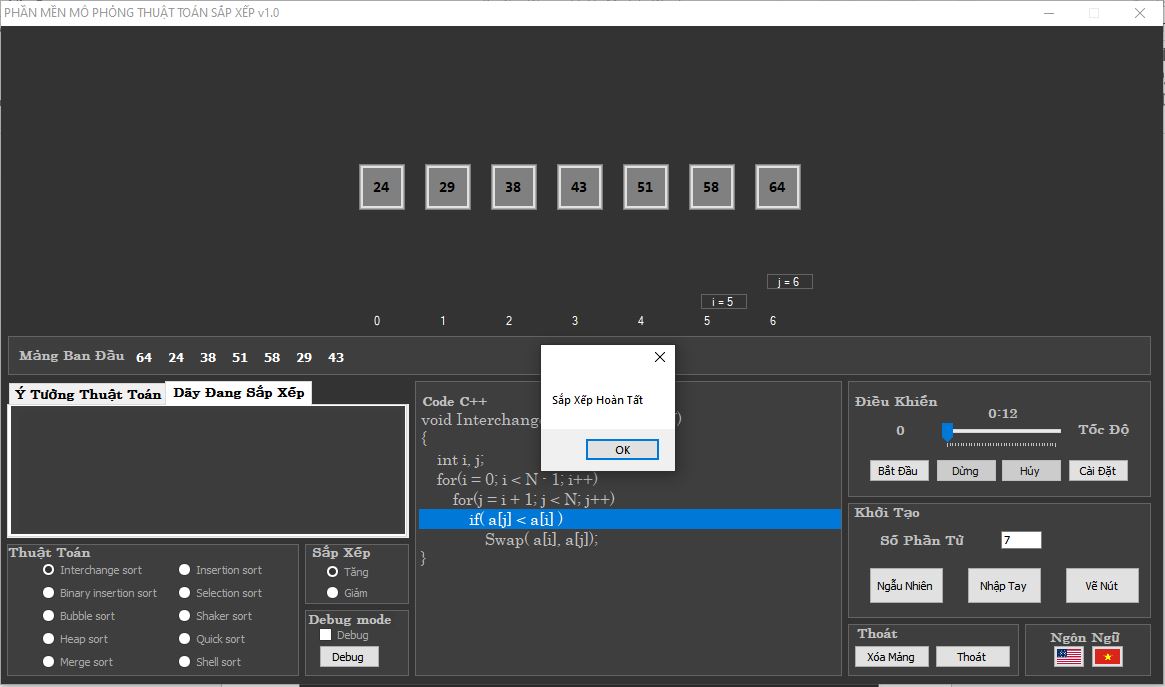


**3.2.1.5. Mô phỏng trên C#**

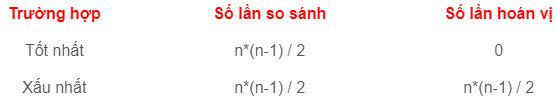
* Người dùng có thể khởi tạo input đầu vào bằng ba cách: Ngẫu nhiên, nhập tay , nhập số phần tử trong textbox và bấm vẽ nút:
* Khi bấm ngẫu nhiên chương trình sẽ tự động cấp phát một mảng ngẫu nhiên n phần tử trong khoản từ 2 đến số lượng node lớn nhấn do người dùng cài đặt, và miền giá trị của từng node vào khoảng từ 1 đến 99.
* Sau đó, người dùng chọn thuật toán Interchange Sort, và có thể chọn kiểu sắp xếp tăng dần hoặc giảm dần, và trong quá trình thực hiện người dùng có thể điều chỉnh tốc độ sắp xếp qua thanh điều khiển và có thể chọn debug chương trình sẽ chạy từng bước của thuật toán để người dùng có thể dễ dàng nắm bắt quá trình.

****

* Đây là mảng sau khi đã sắp xếp xong chương trình, nó sẽ xuất hiện một thông báo mảng đã sắp xếp xong, và hiển thị kết quả sắp xếp thuật toán Interchange Sort.



**3.2.1.6. Đánh giá độ phức tạp**

****

* Độ phức tạp trung bình: O()

**3.2.2. Thuật toán Selection Sort (Sắp xếp chọn trực tiếp)**

**3.2.2.1. Ý tưởng giải thuật**

Chọn phần tử nhỏ nhất trong n phần tử ban đầu, đưa phần tử này về vị trí đúng là đầu tiên của dãy hiện hành. Sau đó không quan tâm đến nó nữa, xem dãy hiện hành chỉ còn n-1 phần tử của dãy ban đầu, bắt đầu từ vị trí thứ 2. Lặp lại quá trình trên cho dãy hiện hành đến khi dãy hiện hành chỉ còn 1 phần tử.

Do dãy ban đầu có n phần tử, vậy tóm tắt ý tưởng thuật toán là thực hiện n-1 lượt việc đưa phần tử nhỏ nhất trong dãy hiện hành về vị trí đúng ở đầu dãy.

**3.2.2.2. Cài đặt mã giả**

* **Bước 1:** i = 0 ;
* **Bước 2**: Tìm phần tử a[min] nhỏ nhất trong dãy hiện hành từ a[i] đến a[n]
* **Bước 3**: Đổi chỗ a[i] và a[min]
* **Bước 4**:

Nếu i < n – 1 thì

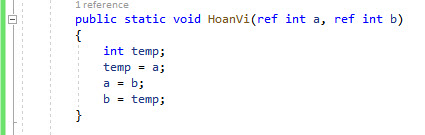
i = i + 1;

Lặp lại bước 2;

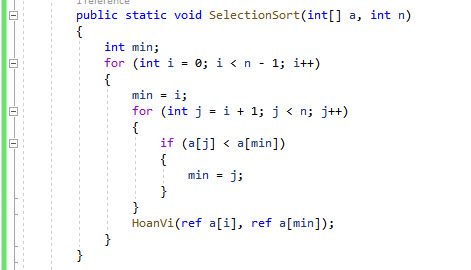
Ngược lại: Dừng

**3.2.2.3. Mô phỏng trên Cửa sổ console**

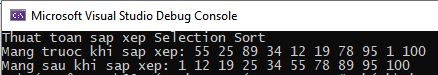
* Hàm hoán vị:



* Hàm sắp xếp Selection Sort

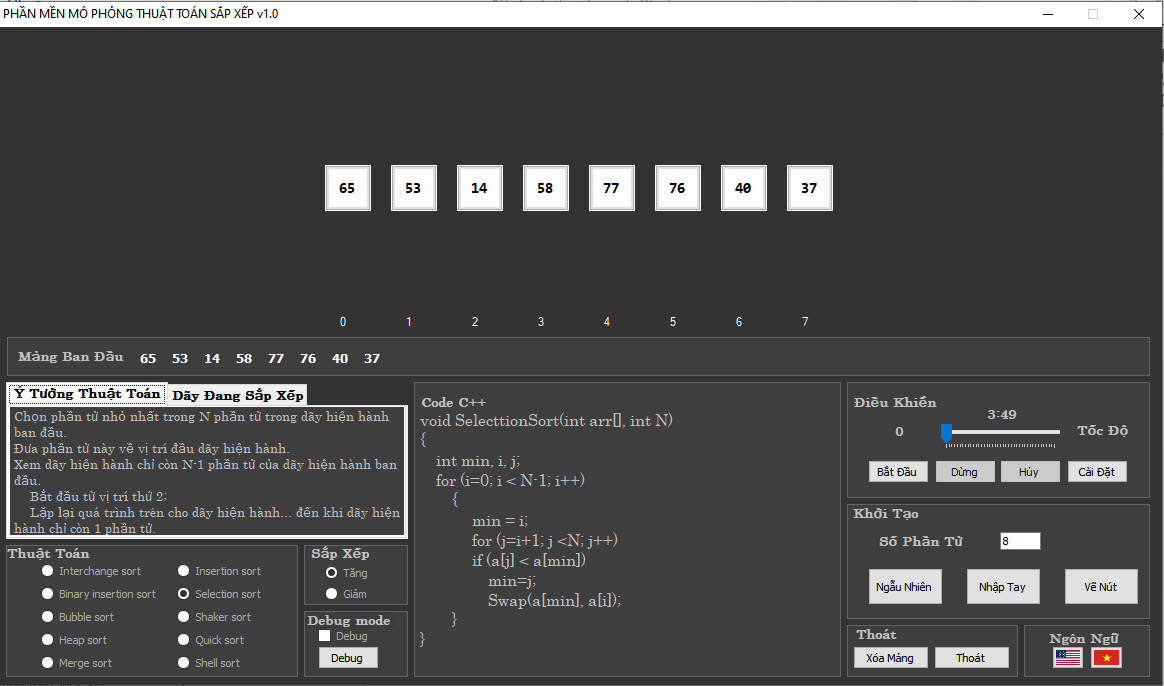


* Kết quả hiển thị trên console:

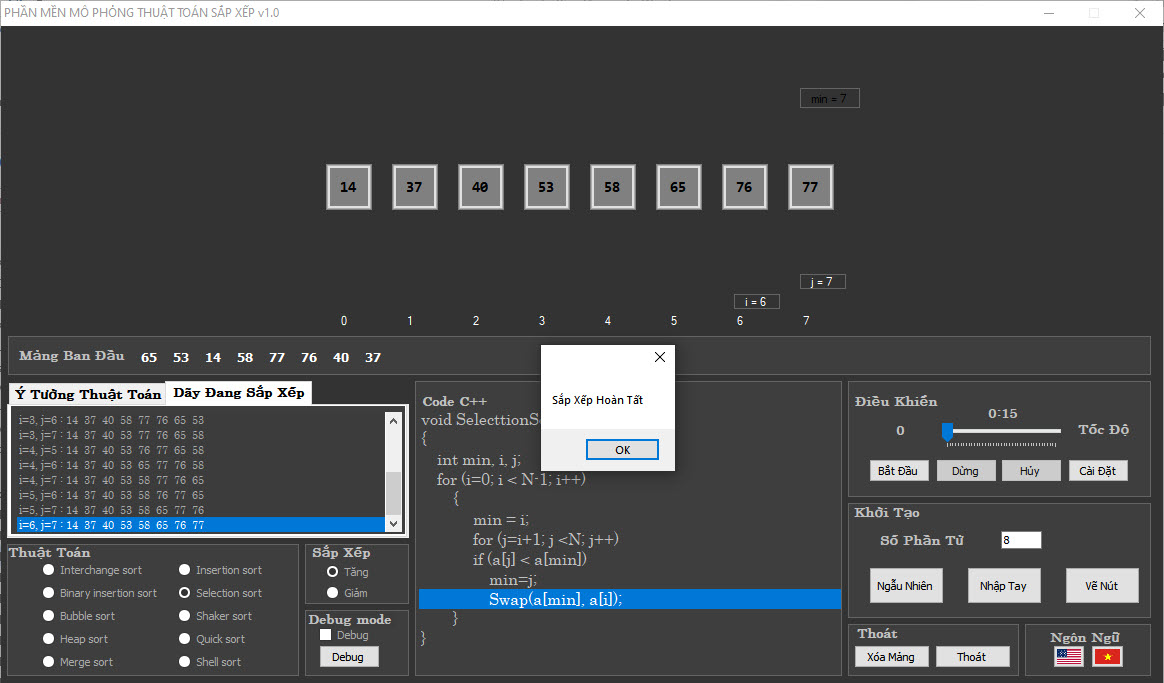


**3.2.2.4. Mô phỏng trên C#**

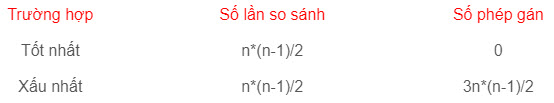
* Ta tạo mới ngẫu nhiên 8 phần tử với giá trị ngẫu nhiên và Check vào thuật toán

****

* Đây là kết quả hiển thị trên màn hình sau khi thực hiện xong việc sắp xếp Selection Sort



**3.2.2.5. Đánh giá độ phức tạp**

****

* Độ phức tạp trung bình : O()

**3.2.3. Thuật toán Bubble Sort (Sắp xếp nổi bọt)**

**3.2.3.1. Ý tưởng thuật toán**

Ý tưởng chính của thuật toán là xuất phát từ cuối (hoặc đầu) dãy, đổi chổ các cặp phần tử kế cận để đưa phần tử nhỏ (lớn) hơn trong cặp phần tử đó về vị trí đúng đầu (cuối) dãy hiện hành, sau đó sẽ không xét đến nó ở vị trí tiếp theo, do vậy ở lần xử lý thứ i sẽ có vị trí đầu dãy là i. Lặp lại xử lý trên cho đến khi không còn cặp phần tử nào để xét.

**3.2.3.2. Cài đặt mã giả**

* **Bước 1**: i=0; //lần xử lý đầu tiên
* **Bước 2**: j = n – 1; //duyệt từ cuối dãy ngược về vị trí i

Trong khi (j > i) thực hiện:

Nếu a[j] < a[j - 1]

HoanVi(a[j], a[j - 1]);

j = j – 1;

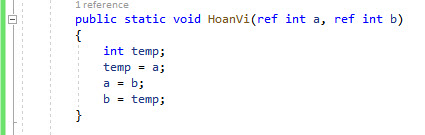
* **Bước 3**: i=i+1; // lần xử lý kế tiếp

Nếu i = n: hết dãy .Dừng

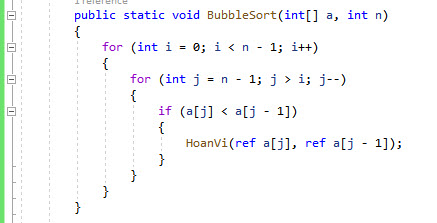
Ngược lại : Lặp lại Bước 2

**3.2.3.3. Mô phỏng trên Cửa sổ console**

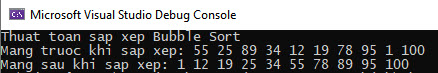
* Hàm hoán vị:



* Hàm sắp xếp Bubble Sort:

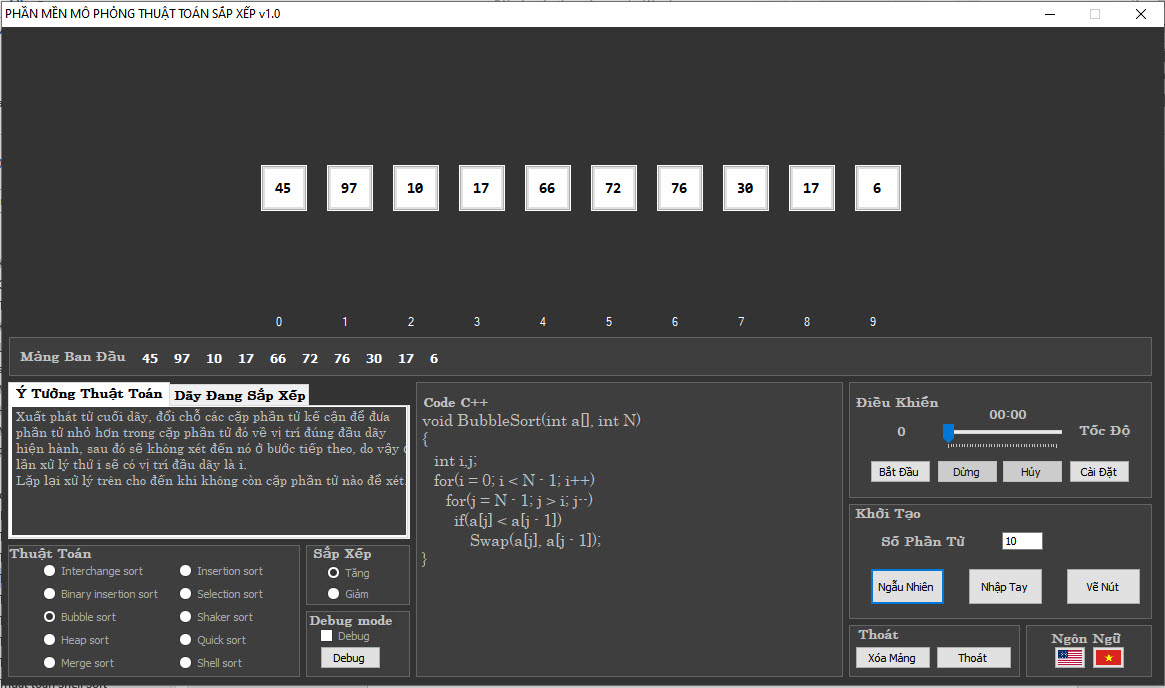


* Kết quả của thuật toán trên console

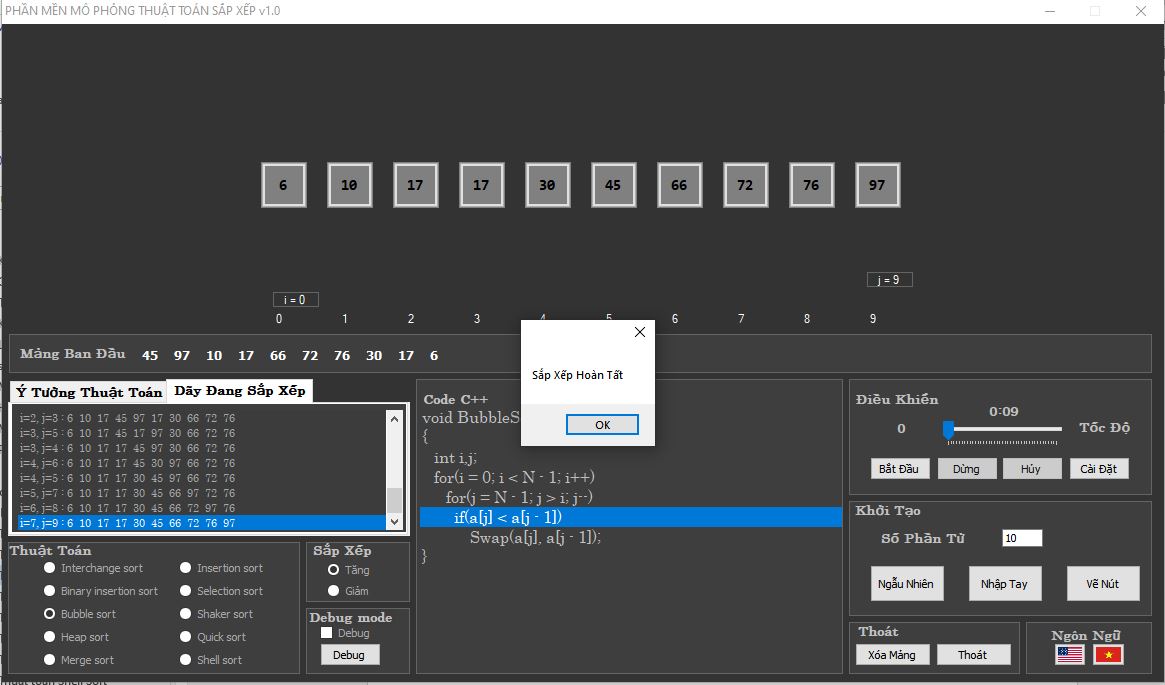


**3.2.3.4. Mô phỏng trên C#**

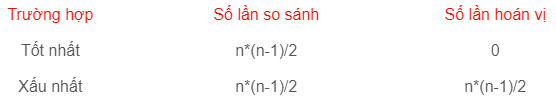
* Ta tạo mới ngẫu nhiên 10 phần tử với giá trị ngẫu nhiên và Check vào thuật toán

****

* Đây là kết quả hiển thị trên màn hình sau khi thực hiện xong việc sắp xếp Bubble Sort



**3.2.3.5. Đánh giá độ phức tạp của thuật toán**



* Độ phức tạp trung bình: O()

**3.2.4. Thuật toán Shaker Sort**

**3.2.4.1.Ý tưởng thuật toán**

Shaker Sort là thuật toán Bubble Sort cải tiến bằng cách thực hiện 2 lượt đi và về cùng 1 lúc về 2 phía khác nhau.

Lượt đi: đẩy phần tử nhỏ về đầu mảng

Lượt về: Đẩy phần tử lớn về cuối mảng

Ghi nhận lại những đoạn chưa sắp xếp nhầm tiết kiệm các phép so sánh thừa

**3.2.4.2. Cài đặt mã giả**

* **Bước 1**: left = 0; right = n-1; // Đoạn left -> right là đoạn cần sắp xếp

k = n; //ghi nhận lại vị trí k xảy ra hoán vị sau cùng để làm cơ sở thu hẹp đoạn left->right

* **Bước *2*:**
* **Bước 2a**:

j = right; //đẩy phần tử nhỏ về đầu mảng

Trong khi j > left

Nếu a[j] < a[j-1] thì {HoanVi(a[j], a[j-1]); k = j;}

j–;

left = k; //loại phần tử đã có thứ tự ở đầu dãy

* **Bước 2b**:

j = left;

Trong khi j < right

Nếu a[j] > a[j+1] thì {HoanVi(a[j], a[j+1]); k = j;}

j++;

right = k; //loại phần tử đã có thứ tự ở cuối dãy

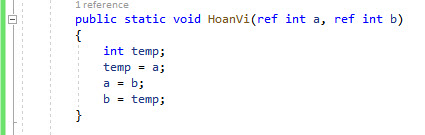
* **Bước 3**:

Nếu left < right lặp lại bước 2

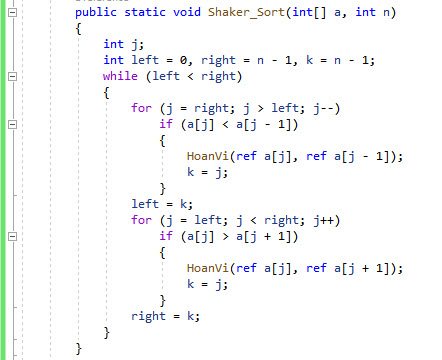
Ngược lại: dừng

**3.2.4.3. Mô phỏng trên Cửa sổ console**

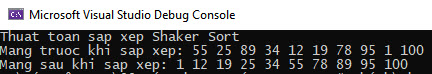
* Hàm hoán vị:



* Hàm sắp xếp Sharker sort:

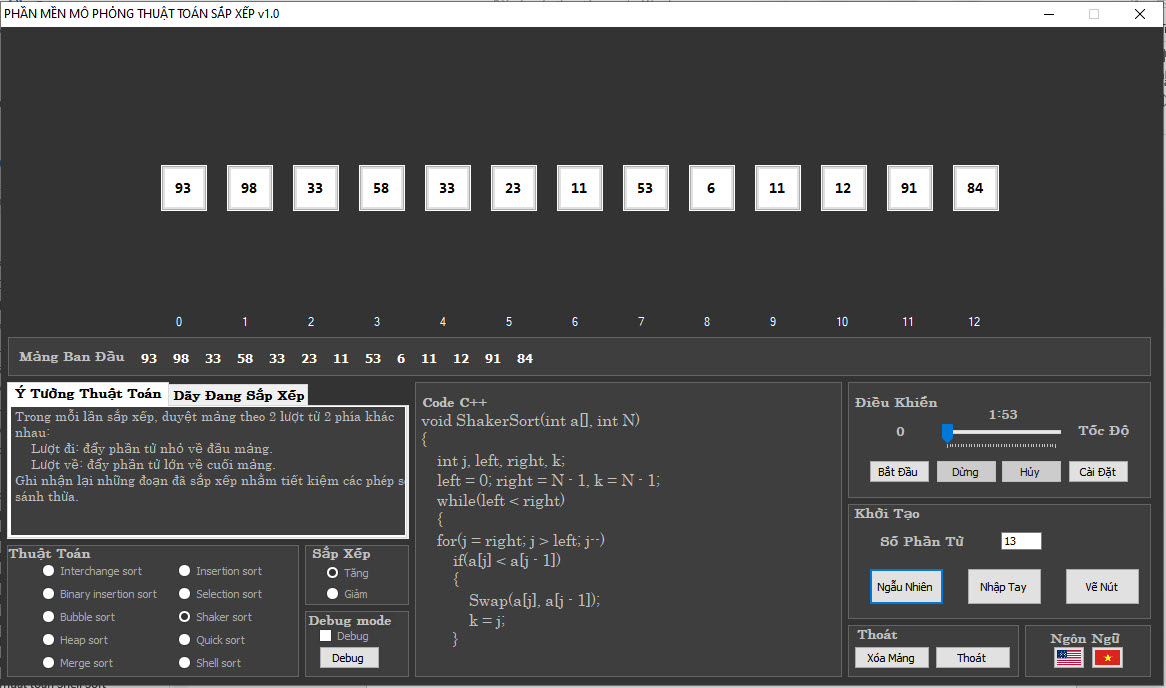


* Kết quả trên cửa sổ console:

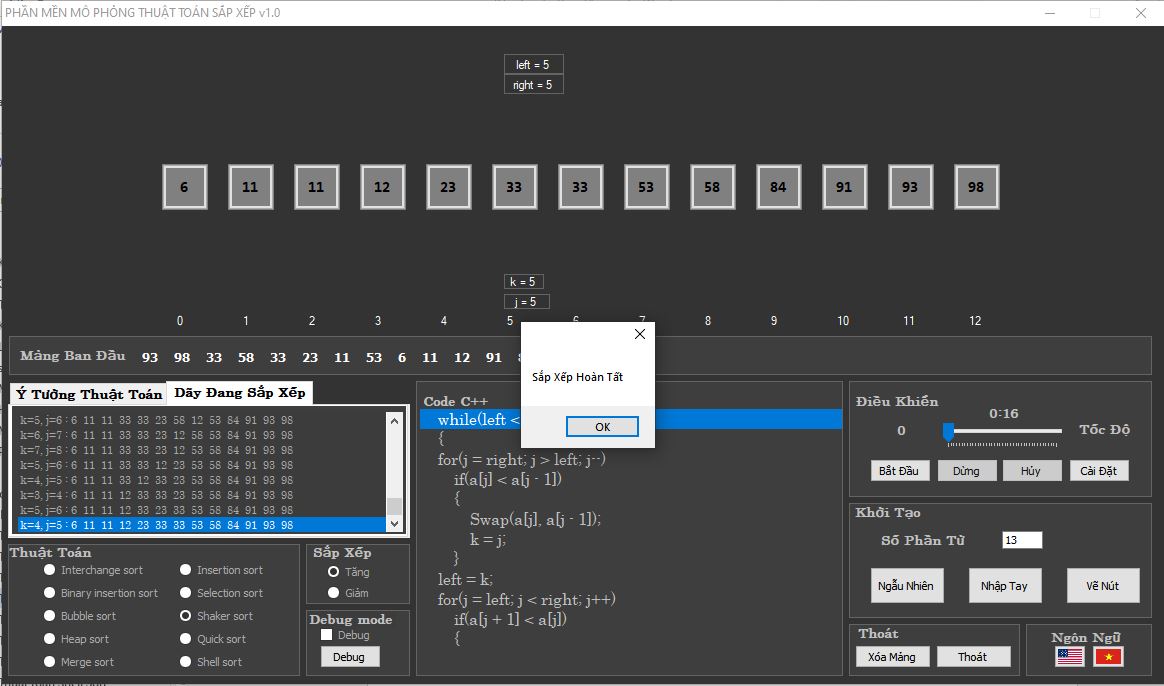


**3.2.4.4. Mô phỏng trên C#**

* Ta tạo mới ngẫu nhiên 13 phần tử với giá trị ngẫu nhiên và Check vào thuật toán

****

* Đây là kết quả hiển thị trên màn hình sau khi thực hiện xong việc sắp xếp Shaker sort



**3.2.4.5. Đánh giá độ phức tạp thuật toán**

****

* Độ phức tạp trung bình: O()

**3.2.5. Thuật toán Insertion Sort (Sắp xếp chèn trực tiếp)**

**3.2.5.1. Ý tưởng thuật toán**

* Giả sử có một mảng **a0, a1, …., an-1** trong đó i phần tử đầu tiên **a0, a1 …,ai-1** đã có thứ tự.
* Tìm cách chèn phần tử ai vào **vị trí thích hợp** của đoạn đã được sắp để có mảng mới a0, a1, …, ai  trở nên có thứ tự. Vị trí này chính là vị trí giữa hai phần tử ak-1 và ak thỏa ak-1 < ai < ak ( 1 )

**3.2.5.2. Cài đặt mã giả**

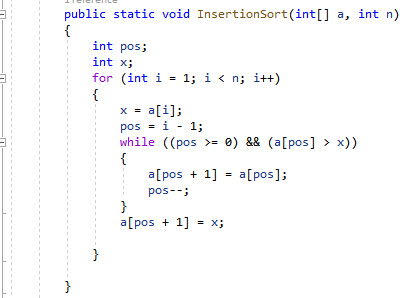
* **Bước 1**: i = 1; // giả sử có đoạn a[1] đã được sắp
* **Bước 2**: x = a[i]; Lưu giá trị a[i] và tìm vị trí pos thích hợp trong đoạn a[1] đến a[i- 1] để chèn a[i] vào
* **Bước 3**: Dời chỗ các phần tử từ a[pos] đến a[i - 1] sang phải 1 vị trí để dành chổ cho a[i]
* **Bước 4**: a[pos] = x; //có đoạn a[1]..a[i] đã được sắp
* **Bước 5**: i = i + 1;

Nếu i < n; Lặp lại Bước 2

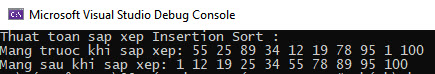
Ngược lại: Dừng

**3.2.5.3. Mô phỏng trên cửa sổ console**

* Hàm sắp xếp Insertion Sort:

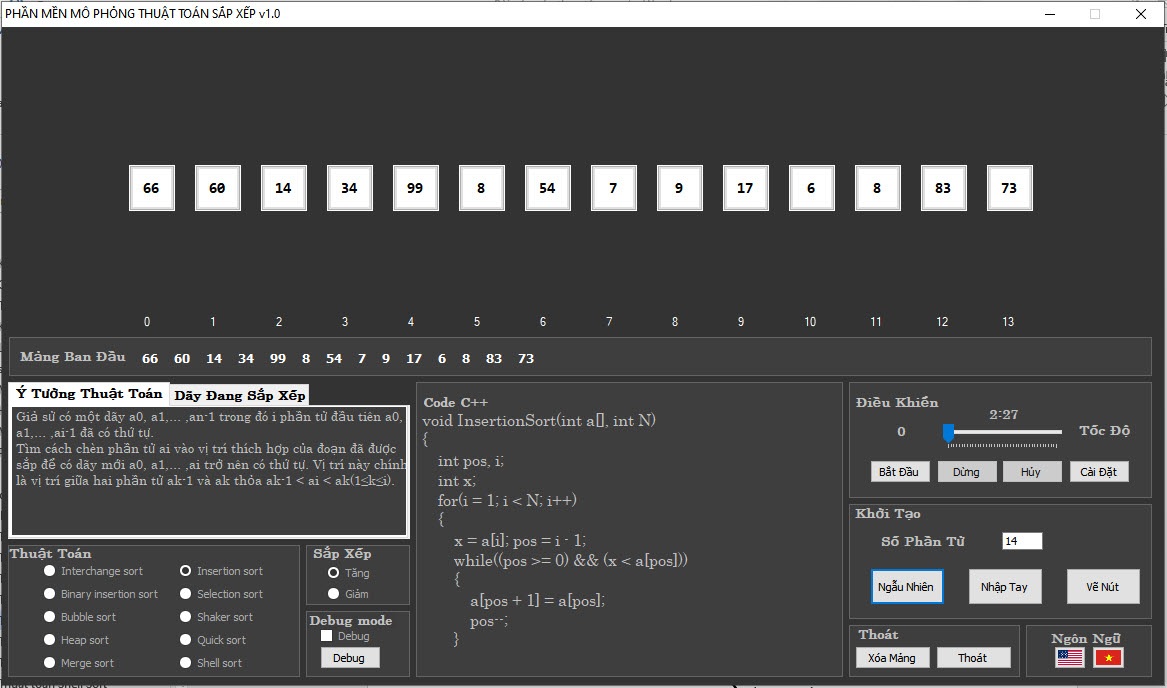


* Kết quả hiển thị trên cửa sổ console

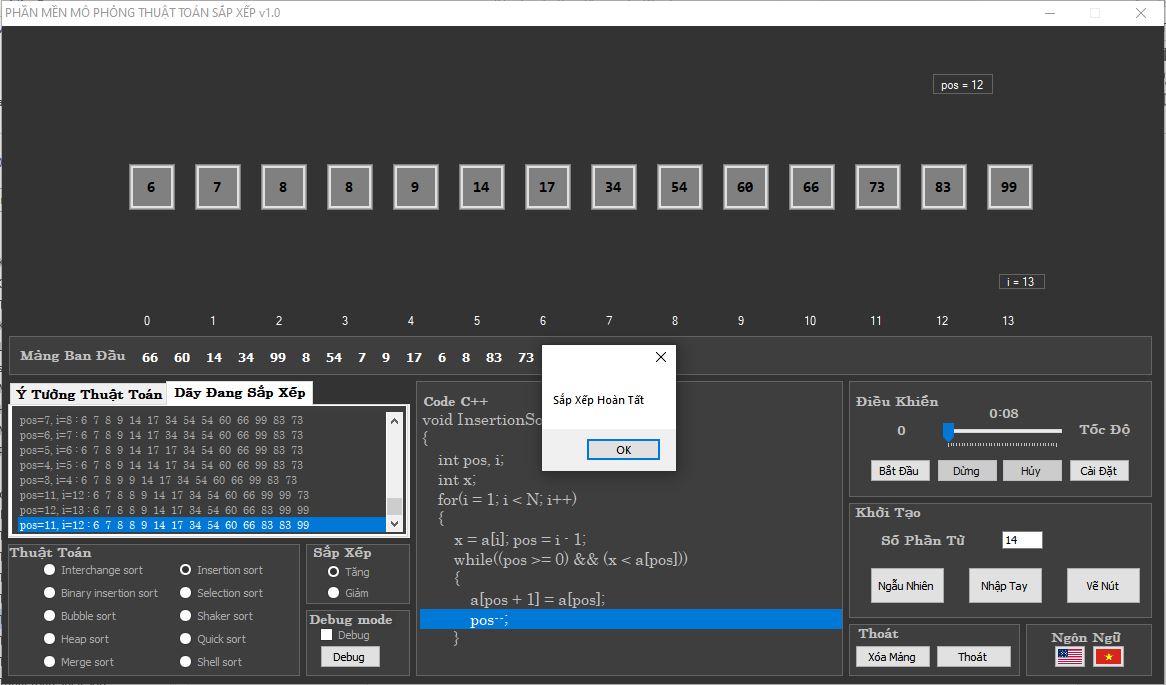


**3.2.5.4. Mô phỏng trên C#**

* Ta tạo mới ngẫu nhiên 14 phần tử với giá trị ngẫu nhiên và Check vào thuật toán

****

* Đây là kết quả hiển thị trên màn hình sau khi thực hiện xong việc sắp xếp Insertion sort



**3.2.5.5. Đánh giá độ phức tạp thuật toán**

* Số phép so sánh
* Tốt nhất
* Xấu nhất:
* Số phép gán:
* Tốt nhất:
* Xấu nhất:
* Độ phức tạp trung bình O()

**3.2.6. Thuật toán Binary insertion Sort(Chèn nhị phân)**

**3.2.6.1. Ý tưởng giải thuật:**

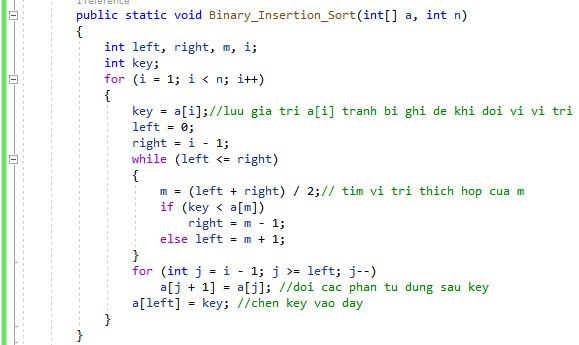
Binary Insertion Sort là cải tiến của Insertion Sort.

Khi tìm vị trí thích hợp để chèn a[i] vào đoạn a[0] đến a[i-1], do đoạn đã được sắp xếp, nên có thể sử dụng giải thuật tìm kiếm nhị phân để thực hiện việc tìm vị trí pos, khi đó có giải thuật sắp xếp chèn nhị phân

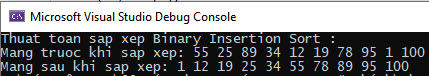
**3.2.6.2. Cài đặt mã giả**

**3.2.6.3. Mô phỏng trên Cửa sổ console**

* Hàm sắp xếp Binary Insertion Sort

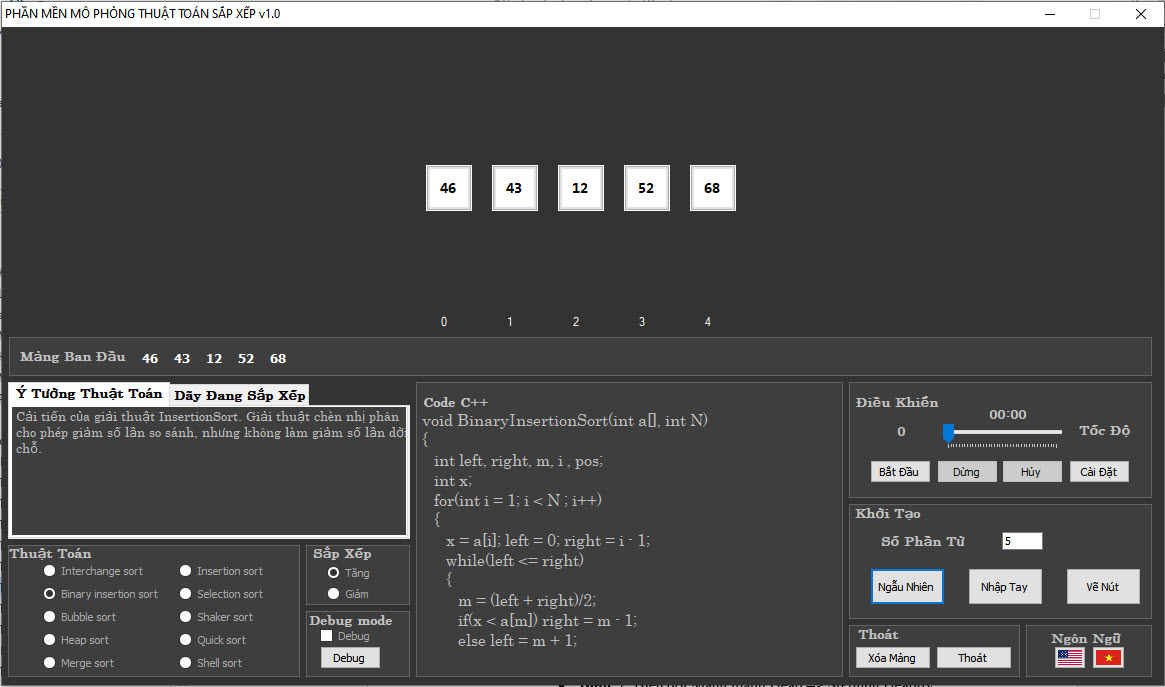


* Kết quả hiển thị trên cửa sổ console:

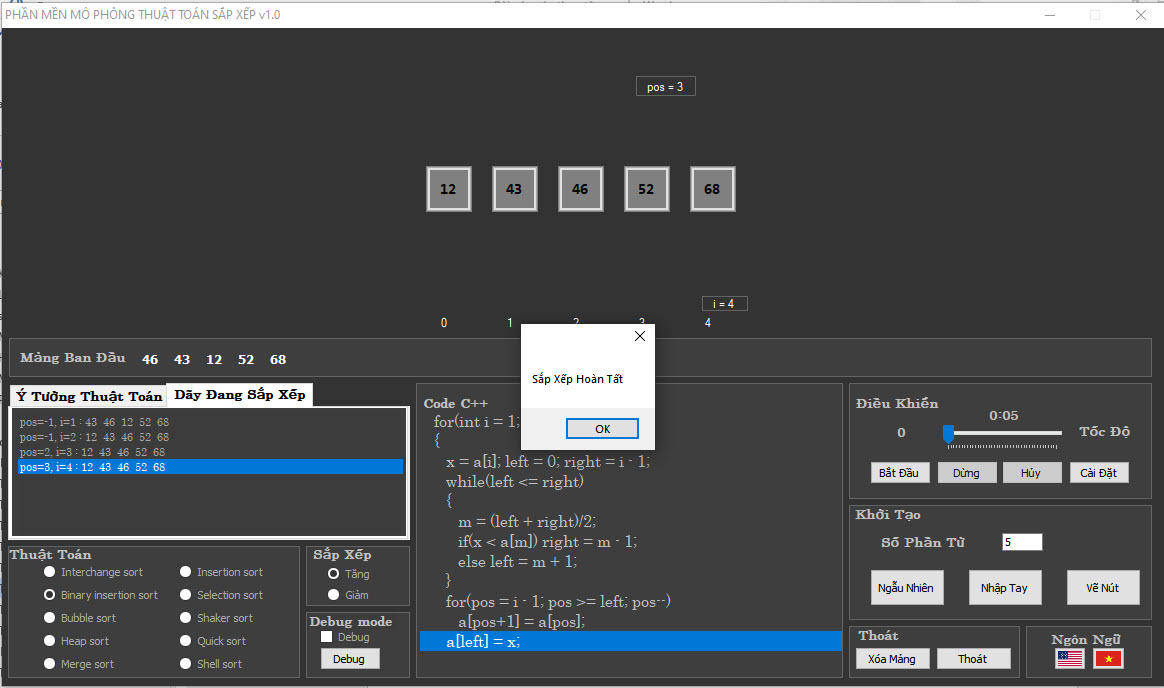


**3.2.6.4. Mô phỏng trên C#**

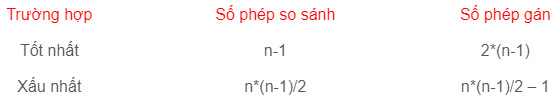
* Ta tạo mới ngẫu nhiên 5 phần tử với giá trị ngẫu nhiên và Check vào thuật toán

****

* Đây là kết quả hiển thị trên màn hình sau khi thực hiện xong việc sắp xếp Binary Insertion sort



**3.2.6.5. Đánh giá độ phức tạp**



* Độ phức tạp trung bình:O()

**3.2.7. Thuật toán Heap Sort (Sắp xếp vun đống)**

**3.2.7.1. Ý tưởng thuật toán**

* **Bước 1**: Xây dựng hàm Heapify để hiệu chỉnh từng nút trên Heap
* **Bước 2**: Biến đổi Mảng thành Heap => Sử dụng Heapify
* **Bước 3**: Hoán vị 2 phần tử đầu và cuối Heap & loại bỏ phần tử cuối
* **Bước 4**: Thực hiện thao tác Heapify để điều chỉnh phần tử đầu tiên

**3.2.7.2. Cài đặt mã giả**

**Giai đoạn 1** : Hiệu chỉnh dãy số ban đầu thành heap

**Giai đoạn 2**: Sắp xếp dãy số dựa trên heap:

* **Bước 1** :Đưa phần tử lớn nhất về vị trí đúng ở cuối dãy:

r = n-1; Swap (a1 , ar );

* **Bước 2**: Loại bỏ phần tử lớn nhất ra khỏi heap: r = r-1;

Hiệu chỉnh phần còn lại của dãy từ a1 , a2 … ar thành một heap

* **Bước 3**:

Nếu r>1 (heap còn phần tử ): Lặp lại Bước 2

Ngược lại : Dừng

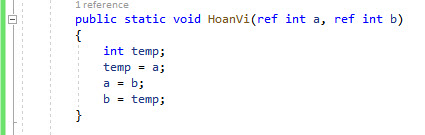
**Heap**:  Là một dãy các phần tử al, al+1 ,… , ar thoả các quan hệ với mọi i ∈ [l,r]:

ai >= a2i+1

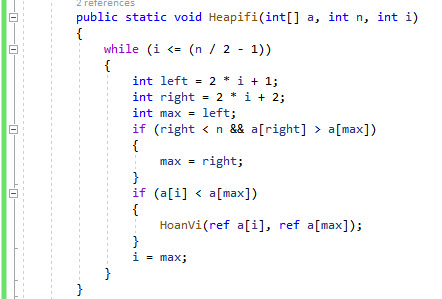
ai >= a2i+2 ((ai; a2i+1), (ai ; a2i+2) là các cặp phần tử liên đới

**3.2.7.3. Mô phỏng trên cửa sổ console**

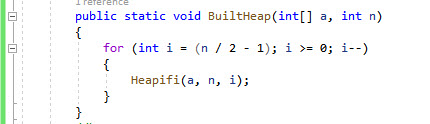
* Hàm hoán vị:



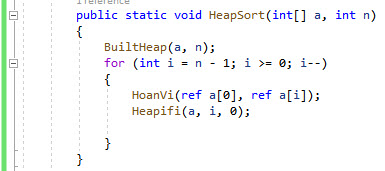
* Hàm Heapify để hỗ trợ tạo cấy heap sắp xếp thứ tự



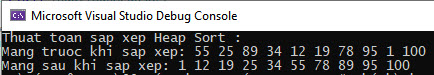
* Hàm BuildHeap để tạp cây



* Hàm sắp xếp HeapSort:

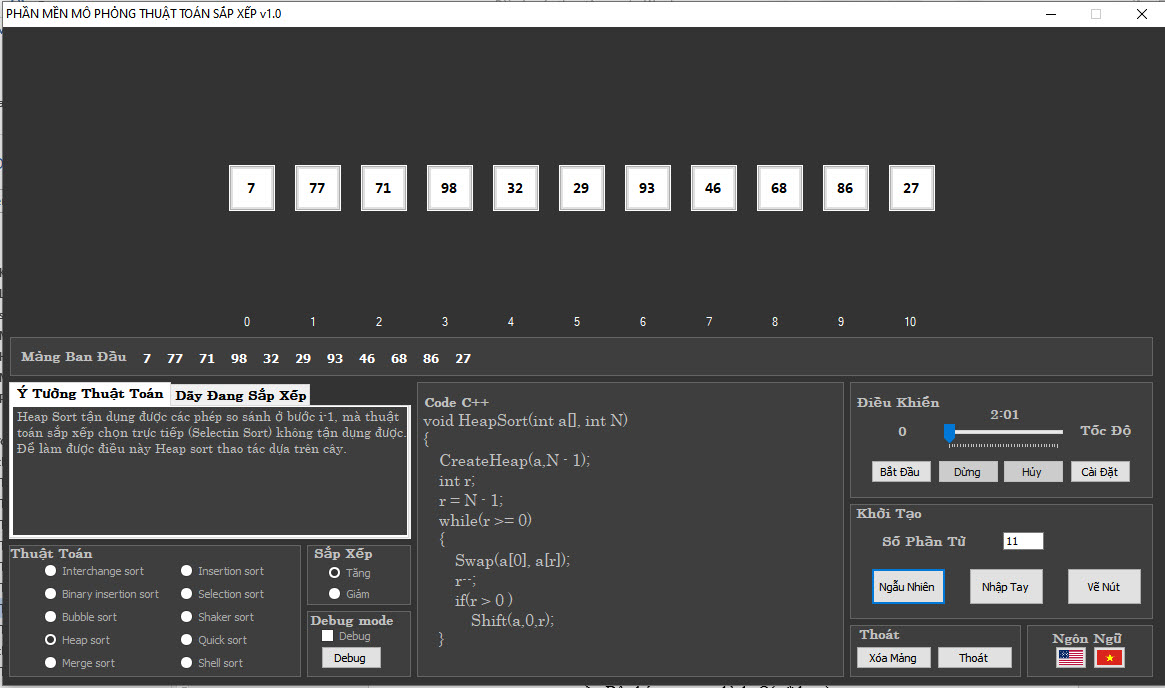


* Kết quả hiển thị trên cửa sổ console:

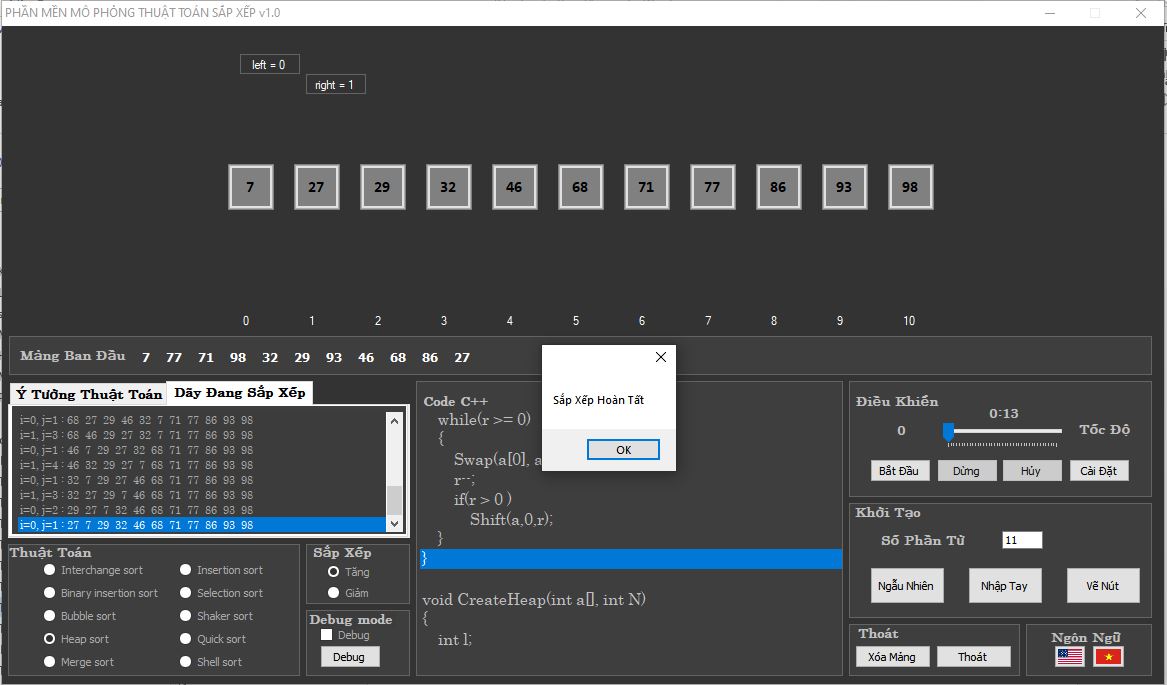


**3.2.7.4. Mô phỏng trên C#**

* Ta tạo mới ngẫu nhiên 11 phần tử với giá trị ngẫu nhiên và Check vào thuật toán

****

* Đây là kết quả hiển thị trên màn hình sau khi thực hiện xong việc sắp xếp Heap sort

****

**3.2.7.5. Đánh giá độ phức tạp thuật toán**

****

* Độ phức tạp trung bình: O(n \* logn)

**3.2.8. Thuật toán Shell Sort**

**3.2.8.1. Ý tưởng thuật toán**

Phân hoạch dãy thành các dãy con. Sắp xếp các dãy con theo phương pháp chèn trực tiếp. Dùng phương pháp chèn trực tiếp để sắp xếp lại cả dãy

Phân chia dãy ban đầu thành những dãy con gồm các phần tử cách nhau **h**vị trí.

Dãy ban đầu: a1, a2, a3, …., an được xem như sự xen kẽ của các dãy con sau:

* Dãy con thứ nhất: a1, ah+1, a2h+,…
* Dãy con thứ hai: a2, ah+2, a2h+2, …
* Dãy con thứ h: ah, a2h, a3h,….

Tiến hành sắp xếp các phần tử trong cùng dãy con sẽ làm cho các phần tử được đưa về vị trí đúng tương đối

Giảm khoảng cách **h** để tạo thành các dãy con mới

Dừng khi h = 1

**3.2.8.2. Cài đặt mã giả**

* **Bước 1***:*Chọn ***k*** khoảng cách h[1], h[2], …. , h[k]

i = 1;

* **Bước 2**:

Phân chia dãy con ban đầu thành các dãy con cách nhau h[i] khoảng cách

sắp xếp từng dãy con bằng phương pháp chèn trực tiếp

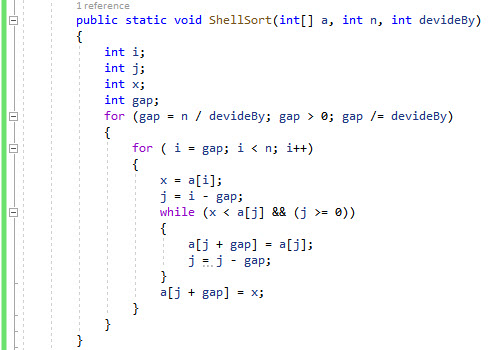
* **Bước 3**: i = i + 1;

Nếu  i > k: dừng

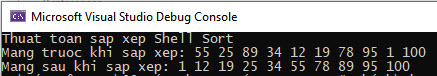
Ngược lại: lặp lại bước 2.

**3.2.8.3. Mô phỏng trên cửa sổ console**

* Hàm sắp xếp Shell Sort:

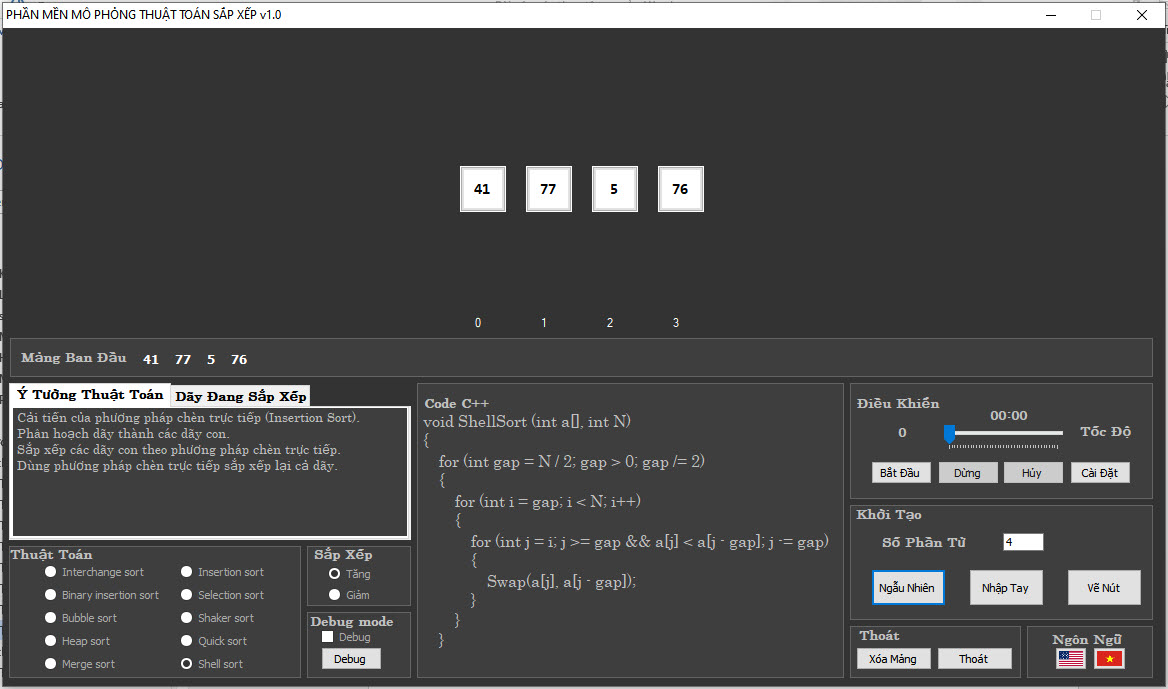


* Kết quả hiển thị trên console:

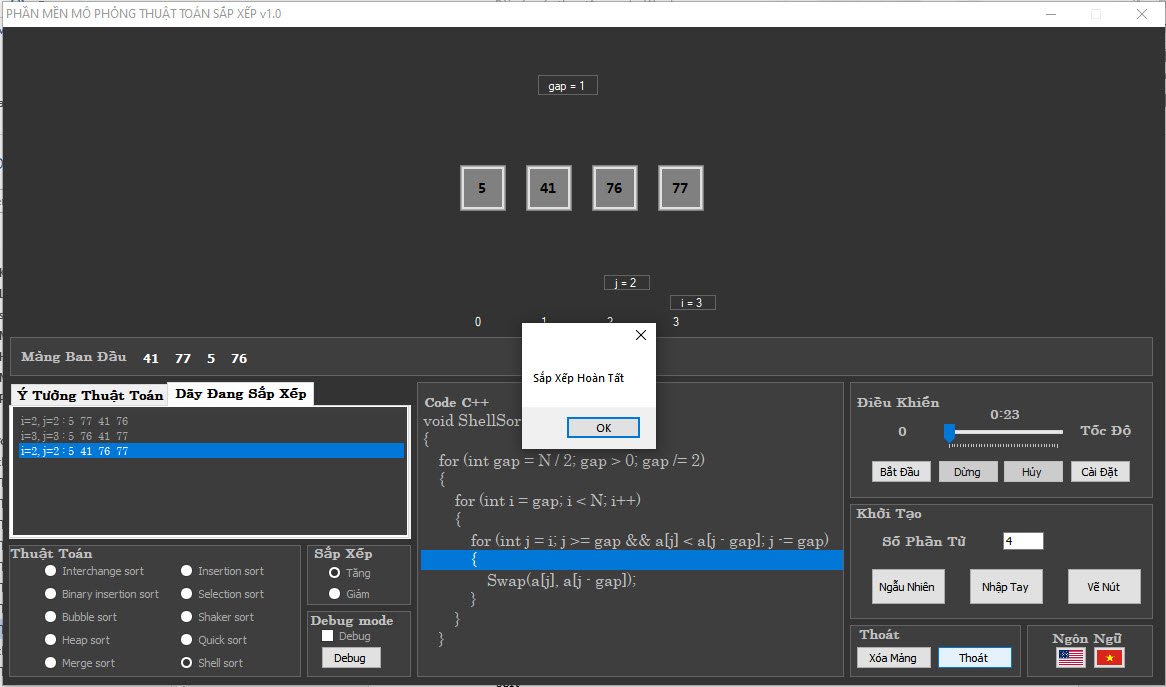


**3.2.8.4. Mô phỏng trên C#**

* Ta tạo mới ngẫu nhiên 4 phần tử với giá trị ngẫu nhiên và Check vào thuật toán

****

* Đây là kết quả hiển thị trên màn hình sau khi thực hiện xong việc sắp xếp Shell sort



**3.2.8.5. Đánh giá độ phức tạp thuật toán**

* Hiện nay đánh giá thuật toán Shell Sort dẫn đến vấn đề toán học rất phức tạp, thậm chí một số chưa được chứng minh.Tuy nhiên, hiệu quả của thuật toán còn phụ thuộc vào dãy các độ dài được chọn.
* Trong trường hợp, chọn dãy độ dài theo công thức:

Hi = (Hi - 1 -1)/2

và Hk = 1, k = log2 – 1

* Độ phức tạp:O(n1.2)
* Độ phức tạp trung bình: O()

**3.3. Các thuật toán sắp xếp sử dụng đệ qui**

**3.3.1. Thuật toán Quick Sort (Sắp xếp nhanh)**

**3.3.1.1. Ý tưởng thuật toán**

Giải thuật Quick sort sắp xếp các dãy a[0], a[1],…., a[n] dựa trên việc phân hoạch dãy ban đầu thành 3 đoạn:

* x **Đoạn 1**: gồm các phần tử có giá trị bé hơn x
* **Đoạn 2**: gồm các phần tử có giá trị bằng x
* **Đoạn 3**: gồm các phần tử có giá trị lớn hơn

x: giá trị của phần tử chính giữa dãy ban đầu.

Do vậy, đoạn thứ 2 đã có sắp xếp.

Để sắp xếp đoạn 1 và đoạn 3, ta lần lượt tiến hành việc phân hoạch từng dãy con theo cùng phương pháp phân hoạch ban đầu vừa trình bày.

**3.3.1.2. Cài đặt mã giả**

* **Bước 1**: Nếu left >= right // dãy có ít hơn 2 phần tử

=> Kết thúc

* **Bước 2**:

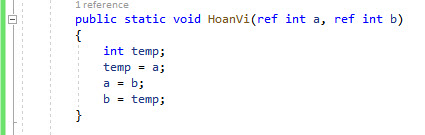
x = a(right – left) /2

Phân hoạch dãy ban đầu aleft…. aright thành các đoạn: aleft….aj, aj+1….ai-1, ai….. aright

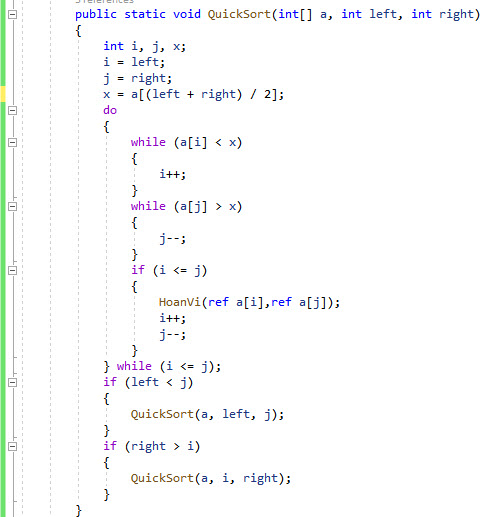
* **Bước 3**: Sắp xếp đoạn 1: aleft….aj
* **Bước 4:** Sắp xếp đoạn 3: ai….. aright

**3.3.1.3. Mô phỏng trên cửa sổ console**

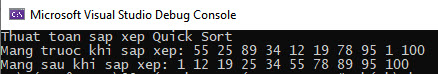
* Hàm hoán vị:



* Hàm sắp xếp Quick Sort:

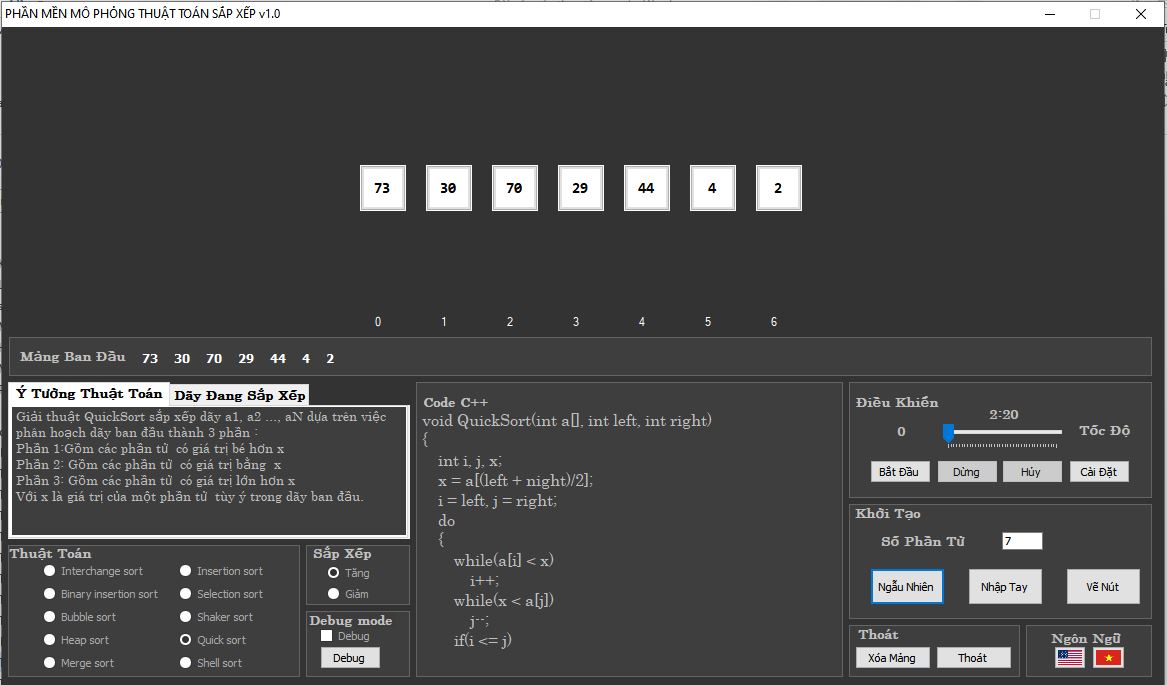


* Kết quả hiển thị trên màn hình console:

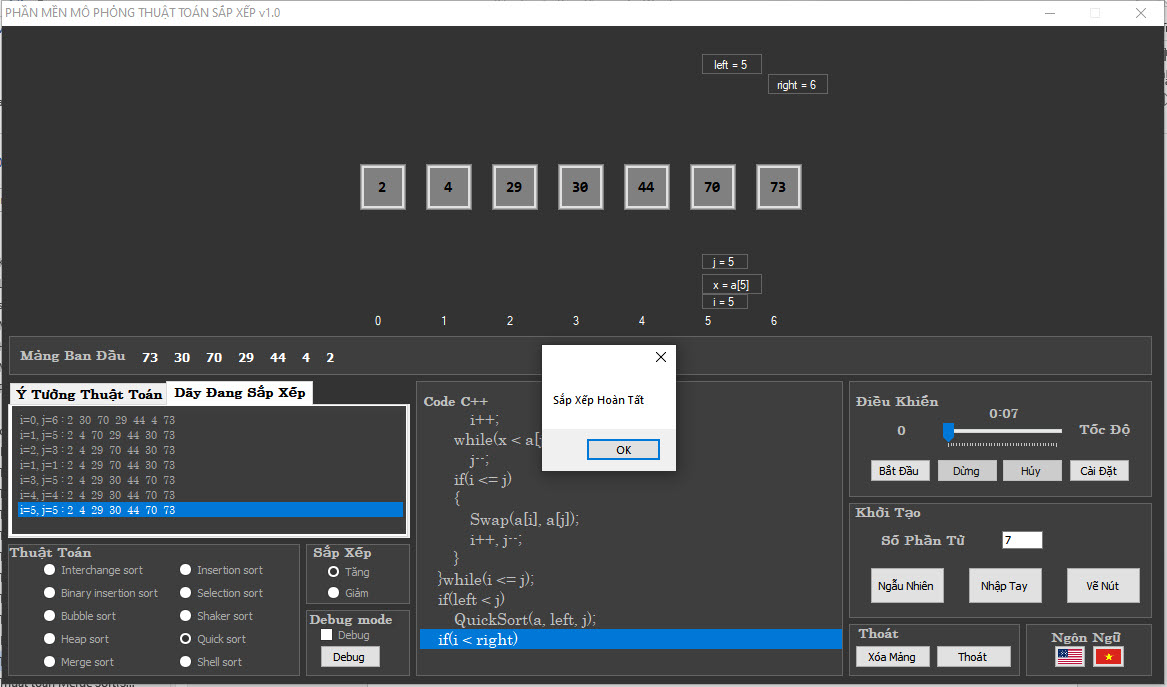


**3.3.1.4. Mô phỏng trên C#**

* Ta tạo mới ngẫu nhiên 7 phần tử với giá trị ngẫu nhiên và Check vào thuật toán

****

* Đây là kết quả hiển thị trên màn hình sau khi thực hiện xong việc sắp xếp Quick sort



**3.3.1.5. Đánh giá độ phức tạp của thuật toán**

****

* Độ phức tạp trung bình: O(nlog(n))

**3.3.2. Thuật toán Merge Sort(Sắp xếp trộn)**

**3.3.2.1. Ý tưởng thuật toán**

Giống như Quick sort, Merge sort là một thuật toán chia để trị. Thuật toán này chia mảng cần sắp xếp thành 2 nửa. Tiếp tục lặp lại việc này ở các nửa mảng đã chia. Sau cùng gộp các nửa đó thành mảng đã sắp xếp. Hàm merge() được sử dụng để gộp hai nửa mảng. Hàm merge(arr, l, m, r) là tiến trình quan trọng nhất sẽ gộp hai nửa mảng thành 1 mảng sắp xếp, các nửa mảng là arr[l…m] và arr[m+1…r] sau khi gộp sẽ thành một mảng duy nhất đã sắp xếp.

**3.3.2.2. Cài đặt mã giả**

* **Bước 1**: // chuẩn bị:

k = 1; //k là chiều dài của dãy con trong bước hiện hành

* **Bước 2** :

Tách dãy Array1, Array2, ., Arraynthành 2 dãy b và c theo nguyên tắc luân phiên từng nhóm k phần tử:

b = Array1, ., Arrayk, Array2k+1, ., Array3k, .

c = Arrayk+1, ., Array2k, Array3k+1, ., Array4k, .

* **Bước 3** :

Trộn từng cặp dãy con gồm k phần tử của 2 dãy temp\_1, temp\_2 vào a.

* **Bước 4 :**

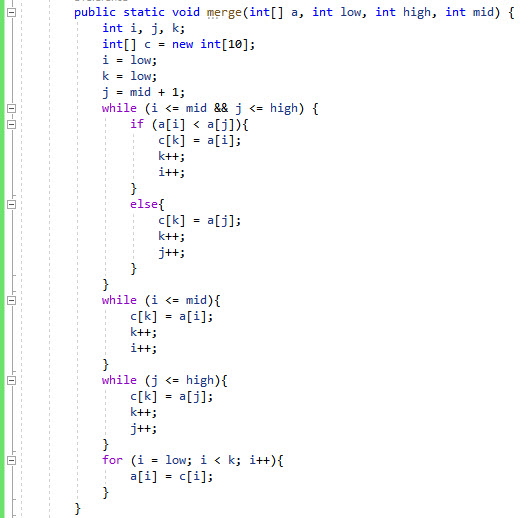
k = k\*2;

Nếu k < n thì trở lại bước 2.

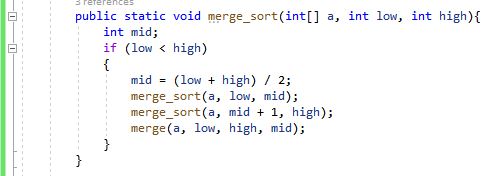
Ngược lại: Dừng

**3.3.2.3. Mô phỏng trên cửa sổ console**

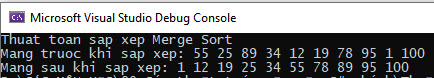
* Hàm merge



* Hàm merge\_sort:

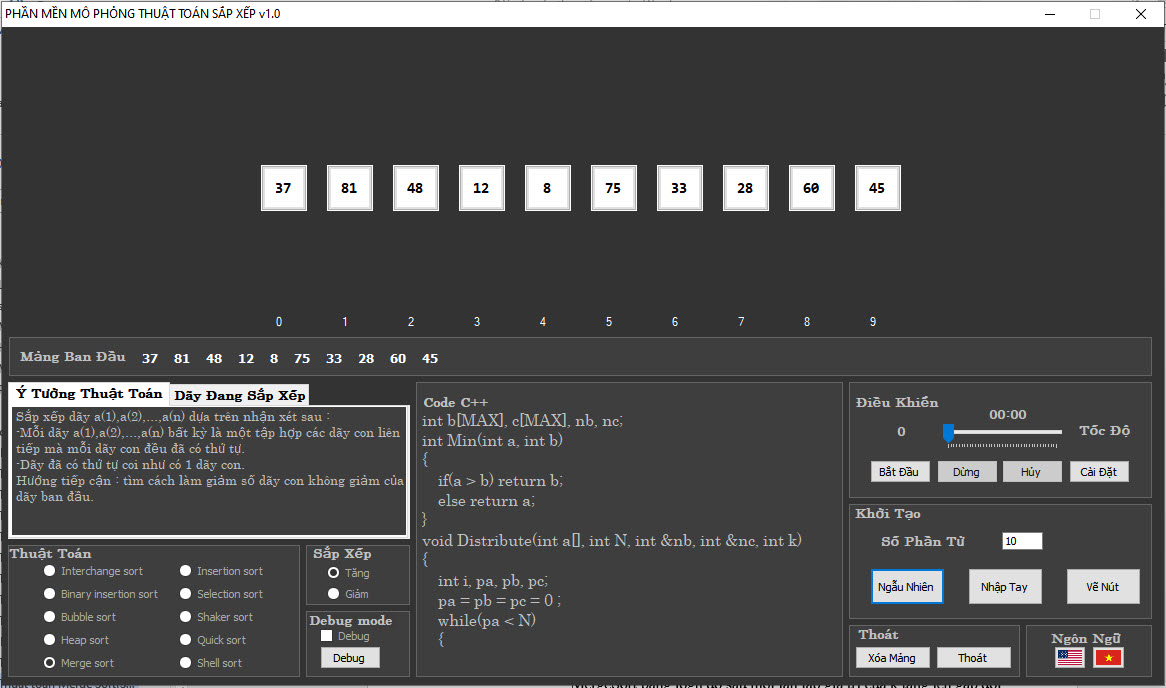


* Kết quả hiển thị trên cửa sổ console

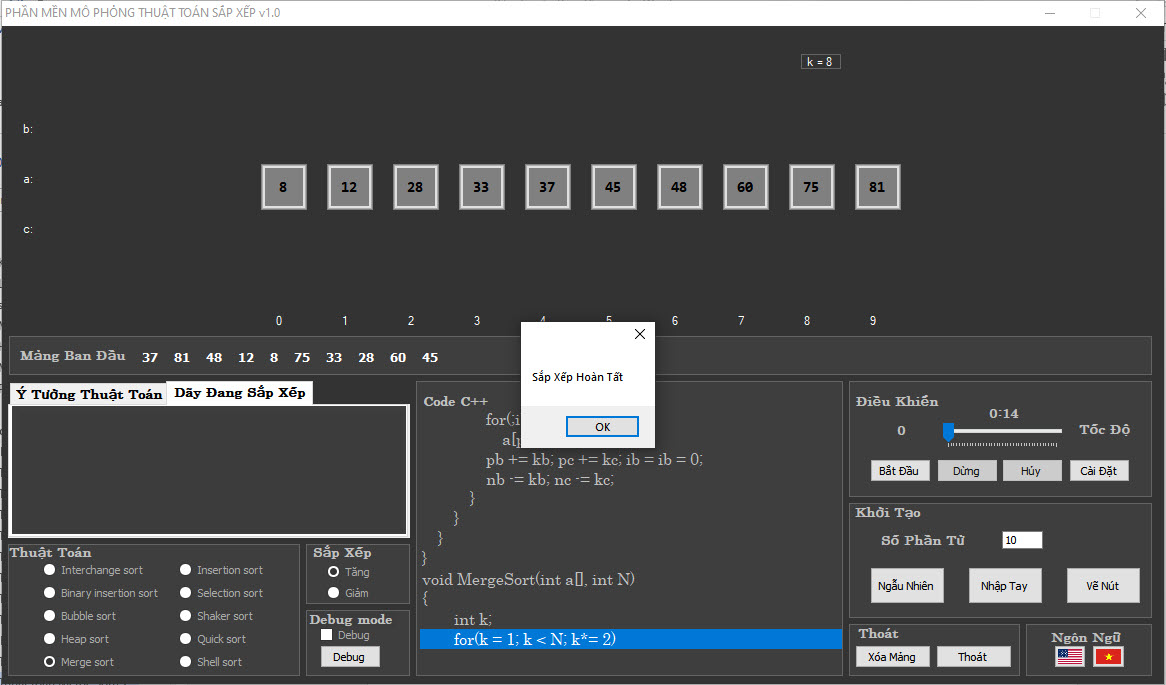


**3.3.2.5. Mô phỏng trên C#**

* Ta tạo mới ngẫu nhiên 10 phần tử với giá trị ngẫu nhiên và Check vào thuật toán

****

* Đây là kết quả hiển thị trên màn hình sau khi thực hiện xong việc sắp xếp Merge sort

****

**3.3.2.6. Đánh giá độ phức tạp của thuật toán**

* Ta thấy rằng số lần lặp của bước 2 và bước 3 trong thuật toán MergeSort bằng logn do sau mỗi lần lặp giá trị của k tăng lên gấp đôi. Dễ thấy, chi phí thực hiện bước 2 và bước 3 tỉ lệ thuận bới n. Như vậy, chi phí  thực hiện của giải thuật MergeSort sẽ là O(nlogn). Do không sử dụng thông tin nào về đặc tính của dãy cần sắp xếp, nên trong mọi trường hợp của thuật toán chi phí là không đổi. Ðây cũng chính là một trong những nhược điểm lớn của thuật toán
* Độ phức tạp trung bình = O(nlogn)

**CHƯƠNG 4. KẾT LUẬN**

4.1 Kết quả đạt được

* Hoàn thành chương trình mô phỏng thuật toán sắp xếp như mục tiêu ban đầu có thể hiện node giữa màn hình sửa node trực tiếp trên màn hình chờ, mô phỏng thành công các thuật toán sắp xếp cùng khung ý tưởng và các tham số hiện thị trên màng hình chờ
* Bước đầu nắm bắt được những kỹ thuật lập trình của C# , winform như timer, multithread, backgoundworker, đa ngôn ngữ sử dụng resources, tự tạo 1 user controls,

**4.2 Hạn chế**

* Giao diện chương trình chưa được đẹp mắt và thân thiện với người dùng
* Xử lý code còn rườm rà
* Số lượng phần tử hiển thị hạn chế
* Gặp một số lỗi khó khắc phục trong quá trình xử lý đa luồng multithread
* Cung cấp ít thuật toán

**4.3 Hướng phát triển**

* Cải thiện giao diện chương trình thân thiện hơn với người sử dụng
* Thêm nhiều chức năng như so sánh độ phức tạp của thuật toán (có animation) từ hai thuật toán trở lên
* Tối ưu hóa hiệu suất chương trình
* Đồng bộ tốc độ xử lý của bộ đếm thời gian và thread
* Thêm nhiều tùy chọn cho người dùng trong phần cài đặt
* Thêm thuật toán cho người dùng lựa chọn
* Thêm nhiều chức năng mới cho phần mềm

**TÀI LIỆU THAM KHẢO**

[1].<https://vi.wikipedia.org/wiki/C%E1%BA%A5u_tr%C3%BAc_d%E1%BB%AF_li%E1%BB%87u>

[2]. <https://vietjack.com/cau-truc-du-lieu-va-giai-thuat/giai-thuat-la-gi.jsp>

[3].<https://vi.wikipedia.org/wiki/C_th%C4%83ng_(ng%C3%B4n_ng%E1%BB%AF_l%E1%BA%ADp_tr%C3%ACnh)>

[4].<https://sites.google.com/site/nthanhbanvl/tin-hoc-lop-10/chuong-i/bai-4-bai-toan-va-thuat-toan>

[5].<http://ect.ictu.edu.vn/attachments/article/215/bai%20giang%20-%20mo%20phong%20va%20mo%20hinh%20hoa_%20duong%20thuy%20huong.pdf>

[6]. <http://ent.htu.edu.vn/tin-tuc-su-kien/mo-phong-thuat-toan-trong-hoc-phan-cau-truc-du-lieu-va-giai-thuat.html>

[7]. <http://doc.edu.vn/tai-lieu/de-tai-mo-phong-mot-so-thuat-toan-do-thi-7228/>

[8].<https://itzone.com.vn/vi/article/gioi-thieu-ve-lap-trinh-c-danh-cho-nguoi-moi-bat-dau/>

[9].<https://vi.wikipedia.org/wiki/Thu%E1%BA%ADt_to%C3%A1n_s%E1%BA%AFp_x%E1%BA%BFp>

[www.Howkteam.vn](http://www.Howkteam.vn)

Timer : <https://msdn.microsoft.com/en-us/library/system.timers.timer(v=vs.110).aspx>

ManualResetevent:<https://msdn.microsoft.com/enus/library/system.threading.manualresetevent(v=vs.110).aspx>

tạo user controls

Task:

https://viblo.asia/p/lap-trinh-bat-dong-bo-trong-c-DZrGNDoWkVB

<https://msdn.microsoft.com/en-us/library/mt472912(v=vs.110).aspx>

Các trang stackoverflow.com và các forum khác.