TRƯỜNG ĐẠI HỌC GIAO THÔNG VẬN TẢI
KHOA CÔNG NGHỆ THÔNG TIN

----o0o----

BÁO CÁO BÀI TẬP LỚN
CẤU TRÚC DỮ LIỆU VÀ GIẢI THUẬT

<u>Đề tài:</u> Sử dụng Vector để quản lý phòng học.
Giải quyết bài toán tính toán và sắp xếp
thông tin gia phả bằng Doubly Linked List.

Giảng viên hướng dẫn:
TS. Hoàng Văn Thông
Sinh viên thực hiện:
Nguyễn Văn Hoàng – 231220789
Lớp: Công Nghệ Thông Tin 2 – K64

Hà Nội, tháng 10 năm 2023

MỤC LỤC

I. GIỚI THIỆU

1. Lý do và động lực

Lý do và động lực để em chọn đề tài về này xuất phát từ mong muốn nắm vững nền tảng quan trọng của lập trình. Vì đây là một lĩnh vực cốt lõi giúp em rèn luyện tư duy giải quyết vấn đề, tối ưu hóa thuật toán, và xây dựng các giải pháp hiệu quả. Em nhận thấy rằng việc hiểu sâu về cấu trúc dữ liệu và giải thuật không chỉ hỗ trợ tốt cho các môn học khác mà còn là một kỹ năng cần thiết để phát triển trong lĩnh vực công nghệ. Bên cạnh đó, sự thách thức và tính ứng dụng cao của đề tài này cũng là động lực lớn để em đầu tư công sức và thời gian nghiên cứu.

2. Định nghĩa bài toán

Bài toán 1: (Phần A) Quản lý phòng học

Input: Gồm các lựa chọn từ 0 - 13 cho phép các thao tác thêm, sửa xóa, sắp xếp… phòng học

Ouput: In ra danh sách phòng học sau khi sắp xếp theo tên, kích cỡ,... , tìm phòng… tùy thiêu yêu cầu người dùng đưa ra

Bài toán 2: (Bài số 31 trong danh sách BTL)

Input:

- Dòng đầu là số bộ test
- Với mỗi bộ test:
  - + Dòng đầu tiên ghi số X (0<X<100) là số người con cháu cần sắp xếp.
  - + Tiếp theo là X dòng, mỗi dòng ghi thông tin về một giấy khai sinh của từng người (thứ tự ngẫu nhiên) gồm 3 thành phần, mỗi thành phần cách nhau một khoảng trống:
    - + Tên người cha: không quá 20 ký tự và không chứa khoảng trống.
    - + Tên người con: không quá 20 ký tự và không chứa khoảng trống.
    - + Tuổi của người cha khi sinh con: 1 số nguyên dương, không quá 100..

Output:

- Với mỗi bộ test, in ra màn hình thứ tự bộ test (xem thêm trong bộ test ví dụ), sau đó lần lượt là từng người trong danh sách tuổi từ cao xuống thấp (không tính cụ Ted). Mỗi người viết ra hai thông tin: tên, một khoảng trống rồi đến tuổi của người đó.
- Nếu hai người có cùng tuổi thì xếp theo thứ tự từ điển.

3. Các nhiệm vụ và mức độ hoàn thành

Nhiệm vụ	Mức độ hoàn thành
- Tìm hiểu lớp vector - Lên ý tưởng bài toán 1 - Thiết kế và cài đặt lớp Room là đại diện cho các phòng học, lớp RoomType là đại diện cho thể loại lớp học. - Thiết kế và cài đặt lớp RoomService để quản lý các phòng học	Đã hoàn thành
- Tìm hiểu danh sách liên kết đôi - Lên ý tưởng bài toán 2 - Triển khai và cài đặt cấu trúc danh sách liên kết đôi - Áp dụng vào bài Tính tuổi và sắp xếp	Đã hoàn thành
- Làm báo cáo - Lưu trữ code bằng Github	Đã hoàn thành

II. PHƯƠNG PHÁP LỰA CHỌN

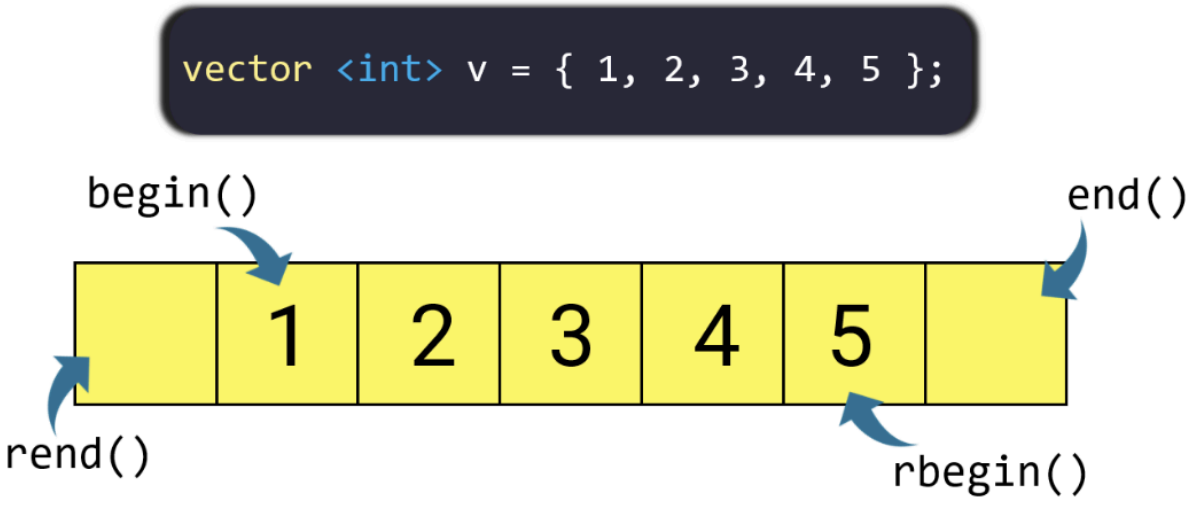
1. Cấu trúc dữ liệu

- VECTOR

Trong C++, vector là một container tuyến tính thuộc thư viện chuẩn (STL - Standard Template Library) và được định nghĩa trong thư viện <vector>. vector là một mảng động, giúp quản lý kích thước và bộ nhớ một cách linh hoạt hơn so với mảng tĩnh. Việc sử dụng nó giải quyết các bài toán thay thế mảng tĩnh là một lựa chọn thích hợp và thường được sử dụng

Đặc điểm của vector

1. Kích thước động: Kích thước của vector có thể thay đổi linh hoạt (tăng hoặc giảm) trong quá trình thực thi chương trình, không giống như mảng tĩnh có kích thước cố định.
2. Quản lý bộ nhớ tự động: vector tự động cấp phát và giải phóng bộ nhớ khi có sự thay đổi về kích thước, giúp lập trình viên giảm thiểu sai sót liên quan đến quản lý bộ nhớ.
3. Truy cập ngẫu nhiên: vector cho phép truy cập phần tử nhanh chóng qua chỉ số, tương tự như mảng.



Hình 1 Minh họa vector và một số phương thức cơ bản

Các thao tác với Vector:

Modifiers

1. push_back(): Hàm đẩy một phần tử vào vị trí sau cùng của vector.

2. assign(): Nó gán một giá trị mới cho các phần tử vector bằng cách thay thế các giá trị cũ.

3. pop_back(): Hàm pop_back () được sử dụng để xóa đi phần tử cuối cùng một vector.

4. insert(): Hàm này chèn các phần tử mới vào trước phần tử trước vị trí được trỏ bởi vòng lặp.

5. erase(): Hàm được sử dụng để xóa các phần tử tùy theo vị trí vùng chứa.

6. swap(): Hàm được sử dụng để hoán đổi nội dung của một vector này với một vector khác cùng kiểu. Kích thước có thể khác nhau.

7. clear(): Hàm được sử dụng để loại bỏ tất cả các phần tử của vùng chứa vector.

Iterators

begin(): đặt iterator đến phần tử đầu tiên trong vector.

end(): đặt iterator đến sau phần tử cuối cùng trong vector.

rbegin(): đặt reverse iterator (trình lặp đảo) đến phần tử cuối cùng trong vector (reverse begin). Nó di chuyển từ phần tử cuối cùng đến phần tử đầu tiên.

rend(): đặt reverse iterator (trình lặp đảo) đến phần tử đầu tiên trong vector (reverse end).

cbegin(): đặt constant iterator (trình vòng lặp) đến phần tử đầu tiên trong vector.

cend(): đặt constant iterator (trình vòng lặp) đến phần tử cuối cùng trong vector.

crbegin(): đặt constant reverse iterator (trình lặp đảo liên tục) đến phần tử cuối cùng trong vector (reverse begin). Nó di chuyển từ phần tử cuối cùng đến phần tử đầu tiên.

crend(): đặt constant reverse iterator (trình lặp đảo liên tục) đến phần tử đầu tiên trong vector.

Capacity

size(): hàm sẽ trả về số lượng phần tử đang được sử dụng trong vector.
max_size(): hàm trả về số phần tử tối đa mà vector có thể chứa.

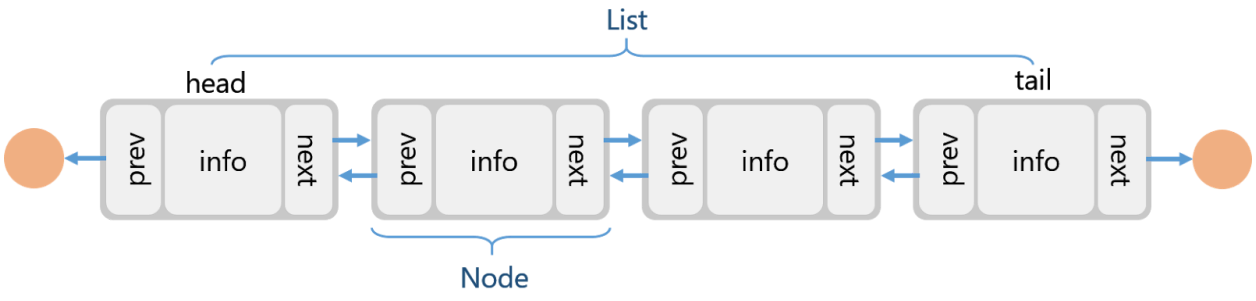
capacity(): hàm trả về số phần tử được cấp phát cho vector nằm trong bộ nhớ.
resize(n): Hàm này thay đổi kích thước vùng chứa để nó chứa đủ n phần tử. Nếu kích thước hiện tại của vector lớn hơn n thì các phần tử phía sau sẽ bị xóa khỏi vector và ngược lại nếu kích thước hiện tại nhỏ hơn n thì các phần tử bổ sung sẽ được chèn vào phía sau vector.
empty(): Trả về liệu vùng chứa có trống hay không, nếu trống thì trả về True, nếu có phần tử thì trả về False.
shrink_to_fit(): Giảm dung lượng của vùng chứa để phù hợp với kích thước của nó và hủy tất cả các phần tử vượt quá dung lượng.
reserve(n): hàm cấp cho vector số dung lượng vừa đủ để chứa n phần tử.

Element access

1. at(g): Trả về một tham chiếu đến phần tử ở vị trí ‘g’ trong vector.
2. data(): Trả về một con trỏ trực tiếp đến (memory array) bộ nhớ mảng được vector sử dụng bên trong để lưu trữ các phần tử thuộc sở hữu của nó.
3. front(): hàm dùng để lấy ra phần tử đầu tiên của vector.
4. back(): hàm dùng để lấy ra phần tử cuối cùng của vector.

- DANH SÁCH LIÊN KẾT ĐÔI (DOUBLY LINKED LIST)

Danh sách liên kết đôi (Doubly Linked List) là một cấu trúc dữ liệu dạng danh sách liên kết, trong đó mỗi nút (node) chứa dữ liệu và hai con trỏ, giúp nó có thể liên kết tới cả nút liền trước và nút liền sau trong danh sách. Đặc điểm của danh sách liên kết đôi là khả năng di chuyển hai chiều: từ đầu danh sách đến cuối danh sách hoặc ngược lại. Vì nó có thể truy xuất phần tử trước và sau nó nên nó là một sự lựa chọn thay thế cho Singly Linked List.



Hình 2 Minh họa danh sách liên kết đôi

Các thao tác trên Doubly Linked List:

Modifiers

1. push_front(): Thêm một phần tử vào đầu danh sách. Khi danh sách rỗng, head và trail đều trỏ đến phần tử mới. Nếu không, phần tử mới trở thành head, và con trỏ prev của head cũ được cập nhật.

2. push_back(): Thêm một phần tử vào cuối danh sách. Khi danh sách rỗng, head và trail đều trỏ đến phần tử mới. Nếu không, phần tử mới được gán làm trail, và con trỏ next của trail cũ được cập nhật.

3. pop_front(): Xóa phần tử đầu tiên của danh sách. Nếu danh sách chỉ có một phần tử, cả head và trail sẽ trở thành nullptr. Nếu không, con trỏ head được cập nhật sang phần tử kế tiếp và phần tử cũ được xóa.

4. pop_back(): Xóa phần tử cuối cùng của danh sách. Nếu danh sách chỉ có một phần tử, head và trail sẽ là nullptr. Nếu không, con trỏ trail được cập nhật và phần tử cũ được xóa.

5. insert(Node<T> *p, T element): Hàm được sử dụng để chèn phần tử mới vào sau phần tử p trong danh sách. Nếu p là nullptr hoặc p là trail, phần tử mới sẽ được thêm vào cuối danh sách. Ngược lại, phần tử mới sẽ được liên kết với p và phần tử tiếp theo của p.

6. erase(Node<T> *p): Xóa phần tử p khỏi danh sách. Nếu p là head, pop_front() được gọi; nếu là trail, pop_back() được gọi. Ngược lại, p sẽ được xóa và các con trỏ next và prev của các phần tử lân cận được cập nhật.

7. sort(bool (*cmp)(T, T)): Sắp xếp các phần tử trong danh sách theo hàm so sánh cmp. Sử dụng hai vòng lặp để so sánh và sắp xếp các phần tử dựa trên hàm cmp.

Iterators

8. begin(): Trả về con trỏ tới phần tử đầu tiên (head) trong danh sách. Dùng để bắt đầu duyệt từ phần tử đầu tiên.

9. end(): Trả về nullptr, đại diện cho phần tử sau phần tử cuối cùng của danh sách. Dùng để kết thúc quá trình duyệt.

10. rbegin(): Trả về con trỏ tới phần tử cuối cùng (tức trail) của danh sách. Dùng để duyệt ngược từ phần tử cuối cùng.

11. rend(): Trả về nullptr, đại diện cho phần tử trước phần tử đầu tiên của danh sách. Dùng để kết thúc quá trình duyệt ngược.

Capacity

12. size(): hàm sẽ trả về số lượng phần tử hiện tại trong danh sách.
13. empty(): Kiểm tra xem danh sách có rỗng không , nếu rỗng thì trả về True, nếu có phần tử thì trả về False.

Element access

14. front(): Trả về tham chiếu đến dữ liệu của phần tử đầu tiên trong danh sách.
15. back(): Trả về tham chiếu đến dữ liệu của phần tử cuối cùng trong danh sách

III. TRIỂN KHAI CÀI ĐẶT

1. Ngôn ngữ lập trình và thư viện

- Ngôn ngữ lập trình: C++
    - Đây là một ngôn ngữ lập trình được phát triển bởi Bjarne Stroustrup vào năm 1979 tại Bell Labs.
    - Tính phổ biến: C++ là một trong những ngôn ngữ lập trình phổ biến nhất trên thế giới, được sử dụng rộng rãi trong nhiều lĩnh vực.
    - Hiệu suất cao: C++ có khả năng thực thi nhanh, thích hợp cho các ứng dụng yêu cầu hiệu suất cao.
    - Thư viện phong phú: Ngôn ngữ C++ cung cấp nhiều thư viện đa dạng, hỗ trợ các tác vụ từ cơ bản đến phức tạp, giúp lập trình viên có nhiều công cụ sẵn có.
    - Đa mô hình: C++ hỗ trợ nhiều mô hình lập trình khác nhau như lập trình cấu trúc, lập trình hướng chức năng, và lập trình hướng đối tượng, giúp lập trình viên linh hoạt chọn cách tiếp cận phù hợp với nhu cầu.
- Thư viện: C++ cung cấp một thư viện chuẩn (STL - Standard Template Library).

2. Tổ chức chương trình và đóng gói

- Tổ chức chương trình:
    - Sử dụng các IDE và Editor: Clion, DevC++, VSCode

    - Chia nhỏ chương trình thành các lớp riêng, hàm, phương thức riêng để dễ đọc, nâng cấp và bảo trì
    - Lưu trữ code ở Github.
- Đóng gói:
    - Các phần code và liên quan đến chương trình đóng gói ở Github (Link: https://github.com/zunohoang/BTL_DSA)
    - Báo cáo: Là file Bao_Cao_BTL_DSA.docx trong link github trên.

IV. PHÂN TÍCH CHƯƠNG TRÌNH

1. Cài đặt chương trình quản lí phòng học:

1.1 Lớp Room

❖ Các thuộc tính:

- id: Mã định danh của phòng.
- name: Tên phòng.
- floor: Tầng của phòng.
- type: Loại phòng (RoomType), có thể là CLASSROOM, LAB, hoặc AUDITORIUM.
- size: Kích thước của phòng.

❖ Các phương thức:

- Constructor:
  + Room(int id, std::string name, int floor, RoomType type, int size): khởi tạo các thuộc tính id, name, floor, type, và size của phòng.
- Getters: Để truy cập các thuộc tính của Room, độ phức tạp của các hàm đều là O(1).
  + getId(): Trả về mã id của phòng.
  + getName(): Trả về tên của phòng.
  + getFloor(): Trả về tầng của phòng.
  + getType(): Trả về loại của phòng.
  + getSize(): Trả về kích thước của phòng.
- Setters: Để cập nhật các thuộc tính của Room, độ phức tạp của các hàm đều là O(1).
  + setId(): Cập nhật mã id của phòng.
  + setName(): Cập nhật tên của phòng.
  + setFloor(): Cập nhật tầng của phòng.
  + setType(): Cập nhật loại của phòng.
  + setSize(): Cập nhật kích thước của phòng.

- roomTypeToString(): chuyển đổi RoomType thành chuỗi để hiển thị với độ phức tạp O(1).

- operator>>(): Nạp chồng toán tử >> để nhập dữ liệu từ người dùng.

- operator<<(): Nạp chồng toán tử << để in thông tin phòng ra màn hình.

1.2 Lớp RoomService:

❖ Các thuộc tính:

- vector<Room> rooms: lưu trữ danh sách các phòng.

❖ Các phương thức:

- Constructor: đầu vào là tên file dữ liệu và nó sẽ lấy thông tin danh sách phòng trong file đấy.

- writeData(): ghi đè lại dữ liệu trong rooms vào file.

- add(): Thêm một phòng vào danh sách rooms với độ phức tạp O(1).

- remove(const int &id): Xóa 1 phòng theo id bằng cách duyệt toàn bộ danh sách và có độ phức tạp O(n).

- showAll(): duyệt qua toàn bộ danh sách và in ra thông tin của mỗi phòng với độ phức tạp O(n).

- showSortBySize(), showSortByName(), showSortByFloor(): sắp xếp danh sách rooms theo thứ tự tăng dần của kích thước, tên, hoặc tầng của phòng với độ phức tạp O(nlogn) của phương thức std::sort().

- showByFloor(): Hiển thị danh sách các phòng nằm ở tầng cụ thể bằng các duyệt qua tất cả các phòng với độ phức tạp O(n)

- showById(): Tìm và hiển thị thông tin phòng dựa trên mã id với độ phức tạp O(n).

- update(): Cập nhật thông tin của phòng dựa trên id với độ phức tạp O(n).

- removeAll(): Xóa tất cả các phòng khỏi danh sách rooms bằng cách gọi rooms.clear(), độ phức tạp O(1).

- showBySize(): Hiển thị tất cả các phòng có kích thước lớn hơn hoặc bằng size, độ phức tạp O(n).

- showByType(): Hiển thị tất cả các phòng có type truyền vào, có độ phức tạp O(n).

- showMenu(): Hiển thị menu chứa các chức năng quản lý phòng.

- program(): Nhận option từ người dùng, thực hiện các chức năng tương ứng. Tùy thuộc vào option, hàm gọi các phương thức như add, remove, showAll, và showByType và xử lý logic để đảm bảo option hợp lệ (0-13).

- run(): Khởi chạy và duy trì chương trình.

2. Cài đặt cấu trúc dữ liệu Doubly Linked List

2.1 Class Node

```
  template <class T>
class Node
{
private:
   T data;
   Node<T> *next;
   Node<T> *prev;

public:
   Node()
   {
      next = NULL;
      prev = NULL;
   }
   Node(T _data)
   {
      this->data = _data;
      next = NULL;
      prev = NULL;
   }
   Node(T _data, Node<T> *_prev, Node<T> *_next)
   {
      this->data = _data;
      this->prev = _prev;
      this->next = _next;
   }
   Node<T> *getNext()
   {
      return next;
   }
   Node<T> *getPrev()
   {
      return prev;
   }
```

```
    T &getData()
    {
        return data;
    }
    void setNext(Node<T> *_next)
    {
        this->next = _next;
    }

    void setPrev(Node<T> *_prev)
    {
        this->prev = _prev;
    }
    void setData(T _data)
    {
        this->data = _data;
    }
};
```

2.2 Phương thức khởi tạo

```
    Doubly_Linked_List()
    {
        head = NULL;
        trail = NULL;
        n = 0;
    }
```

2.3 Bộ lặp xuôi và bộ lặp ngược

Bộ lặp xuôi

```
template <class T>
class DList_Iterator
{
    Node<T> *curr;
public:
    DList_Iterator(Node<T> *c = 0)
    {
```

```
    curr = c;
  }

  Node<T> *getcurr()
  {
    return curr;
  }

  DList_Iterator &operator=(DList_Iterator<T> it)
  {
    this->curr = it.getcurr();
    return *this;
  }

  bool operator!=(DList_Iterator<T> it)
  {
    return curr != it.getcurr();
  }

  T &operator*()
  {
    return curr->getData();
  }

  DList_Iterator operator++(int)
  {
    DList_Iterator it = curr;
    curr = curr->getNext();
    return it;
  }
  DList_Iterator operator++()
  {
    curr = curr->getNext();
    return curr;
  }
};
```

Bộ lặp ngược

```
template <class T>
class DList_Reverse_Iterator
```

```
{
   Node<T> *curr;

public:
   DList_Reverse_Iterator(Node<T> *c = 0) { curr = c; }

   Node<T> *getcurr() { return curr; }

   DList_Reverse_Iterator &operator=(DList_Reverse_Iterator<T> it)
   {
      this->curr = it.getcurr();
      return *this;
   }

   bool operator!=(DList_Reverse_Iterator<T> it)
   {
      return curr != it.getcurr();
   }

   T &operator*()
   {
      return curr->getData();
   }

   DList_Reverse_Iterator operator++(int)
   {
      DList_Reverse_Iterator it = curr;
      curr = curr->getPrev();
      return it;
   }

   DList_Reverse_Iterator operator++()
   {
      curr = curr->getPrev();
      return curr;
   }
};
```

2. 4. Bổ sung phương thức sort

```
void sort(bool (*cmp)(T, T))
{
    for (auto i = head; i != nullptr; i = i->getNext())
    {
        for (auto j = i->getNext(); j != nullptr; j = j->getNext())
        {
            if (cmp(i->getData(), j->getData()))
            {
                T temp = i->getData();
                i->setData(j->getData());
                j->setData(temp);
            }
        }
    }
}
```

- Ở đây thì truyền vào hàm cmp ví dụ cách sử dụng như sau, truyền vào một hàm để định nghĩa như thế nào thì đổi chỗ hai phần tử, sử dụng sắp xếp nổi bọt

```
students.sort(
          [](Student a, Student b){
        return a.getAge() < b.getAge();
          }
);
```

2. 5 Các phương thức khác (Capacity, Modifier, Elements access)
Code đầy đủ có thể xem đầy đủ ở Github.

Link: [zunohoang/BTL_DSA: BTL_DSA]

3. Cài đặt thuật toán tính và sắp xếp thông tin gia phả

3.1 Struct Person:

```
struct Person
```

```
{
   string parent;
   string name;
   int age;
   int parent_age_when_born;
};
```

Person lưu trữ thông tin của mỗi cá nhân bao gồm:

- parent: Tên cha/mẹ của cá nhân.
- name: Tên của cá nhân.
- age: Tuổi của cá nhân (khởi tạo là -1 nếu chưa tính được).
- parent_age_when_born: Tuổi của cha/mẹ khi sinh cá nhân.

3.2 Class ProgramService

```
class ProgramService
{
private:
   Doubly_Linked_List<Person> people;

public:
   void input()
   {
      int n;
      cin >> n;

      for (int i = 0; i < n; i++)
      {
         Person person;
         cin >> person.parent;
         cin >> person.name;
         cin >> person.parent_age_when_born;
         person.age = -1;
         people.push_back(person);
      }

      people.push_back({"", "Ted", 100, 0});
   }
   void run()
   {
```

```
    bool allAgesCalculated = false;
    while (!allAgesCalculated)
    {
        allAgesCalculated = true;

        for (auto &person : people)
        {
            if (person.age == -1)
            {
                bool foundParentWithAge = false;

                for (const auto &parent : people)
                {
                    if (person.parent == parent.name && parent.age != -1)
                    {
                        person.age = parent.age - person.parent_age_when_born;
                        foundParentWithAge = true;
                        break;
                    }
                }

                if (!foundParentWithAge)
                {
                    allAgesCalculated = false;
                }
            }
        }
    }

    people.sort(cmp);
    people.pop_front();
    for (const auto &person : people)
    {
        cout << person.name << " " << person.age << endl;
    }
}

static bool cmp(Person a, Person b)
{
```

```
        if (a.age == b.age)
        {
            return a.name < b.name;
        }
        return a.age < b.age;
    }
};
```

- Hàm input():

  - Nhận số lượng người dùng n và nhập thông tin cho từng người.

  - Thêm mỗi Person vào danh sách people với age mặc định là -1.

  - Thêm thông tin của "Ted" với tuổi được thiết lập cố định là 100, và không có cha (trường parent là chuỗi rỗng ""). "Ted" là điểm khởi đầu để tính tuổi các thành viên khác.

- Hàm run():

  - Sử dụng vòng lặp while để duyệt danh sách cho đến khi mọi người đều có tuổi (age != -1). Với mỗi cá nhân chưa tính được tuổi (có age = -1), chương trình tìm kiếm cha mẹ của họ trong danh sách. Nếu tìm thấy cha mẹ có tuổi đã biết, tuổi của cá nhân được tính dựa trên công thức: person.age = parent.age - person.parent_age_when_born

  - Sắp xếp danh sách people theo thứ tự giảm dần của tuổi. Nếu hai người có cùng tuổi, chương trình sắp xếp theo tên (hàm cmp được sử dụng để so sánh).

  - Sau khi sắp xếp, chương trình loại bỏ "Ted" khỏi danh sách bằng pop_front() và in kết quả ra màn hình.

- Hàm so sánh cmp() :

  Hàm cmp là hàm so sánh cho phép sắp xếp Person trong danh sách:

  - Nếu hai người có cùng tuổi, so sánh theo tên.

  - Nếu không, sắp xếp theo tuổi theo thứ tự tăng dần (để hàm sort có thể đảo ngược thành thứ tự giảm dần).

V. KẾT QUẢ THỰC NGHIỆM

1. Các kết quả

- Các chức năng trong chương trình:
    - Phần A (Bài toán quản lý phòng học):
        - Các chức năng tương đã thử nghiệm nhiều lần và cho ra kết quả tốt về mặt chính xác, tốc độ.
        - Bám sát các yêu cầu mà đề bài đề ra.
        - Dữ liệu sẽ được đọc và lưu trữ ở file.
        - Một số chức năng như:
            - 1. Thêm phòng.
            - 2. Xóa phòng.
            - 3. Hiển thị danh sách phòng.
            - 4. In danh sách phòng sắp xếp theo kích cỡ.
            - 5. In danh sách phòng sắp xếp theo tên.
            - 6. In danh sách phòng sắp xếp theo tầng.
            - 7. In danh sách phòng theo tầng.
            - 8. Tìm phòng theo mã phòng.

    - 9. Cập nhật thông tin phòng.
    - 10. Xóa tất cả các phòng.
    - 11. Tìm phòng có kích cỡ lớn hơn hoặc bằng.
    - 12. Tìm và in danh sách phòng theo loại phòng.
    - 13. Hiển thị lại menu.
    - 14. Thoát chương trình.
- Phần B (Bài 31. Tính toán con cháu cụ Ted) :
    - Thử nghiệm trên bộ test đề bài và tự tạo thêm một số bộ test bằng tay đem ra kết quả chính xác tuyệt đối trong giới hạn mà đề bài đem ra.

- **Kết quả phần A:**

- Hiển thị menu danh sách các chức năng

```
+----CHAO MUNG BAN DEN VOI QUAN LY PHONG HOC----+
De tiep tuc an phim Enter
============================MENU_CRUD============================
==                                                             ==
==     1. Them phong.                                          ==
==     2. Xoa phong                                            ==
==     3. Hien danh sach phong                                 ==
==     4. In danh sach phong sap xep theo kich co              ==
==     5. In danh sach phong sap xep theo ten                  ==
==     6. In danh sach phong sap xep theo tang                 ==
==     7. In danh sach theo tang                               ==
==     8. Tim phong co theo ma phong                           ==
==     9. Cap nhat thong tin phong                             ==
==     10. Xoa tat ca cac phong                                ==
==     11. Tim phong co kich co lon hon hoac bang              ==
==     12. Tim va in danh sach theo loai phong                 ==
==     13. Hien thi lai MENU                                   ==
==     0. Thoat                                                ==
==                                                             ==
========================END_CRUD=============================
Vui long nhap [0-13]:
```

- <u>Chức năng thêm phòng mới:</u>

```
Vui long nhap [0-13]: 1
--THEM PHONG--
Moi nhap thong tin phong
Moi nhap ID: 401
Moi nhap ten: Phong401
Moi nhap tang: 4
Chon loai phong:
   1. Phong hoc
   2. Phong thi nghiem
   3. Phong hoi thao
Nhap (1/2/3):
1
Moi nhap suc chua phong: 32
Them phong thanh cong!


Thuc hien chuc nang khac vui long chon [0-13] hoac chá»n 13 de hien MENU
Vui long nhap [0-13]:
```

- Xóa phòng:

```
Thuc hien chuc nang khac vui long chon [0-13] hoac chá»n 13 de hien MENU
Vui long nhap [0-13]: 2
--XOA PHONG--
Nhap ma phong can xoa: 101
Xoa phong thanh cong!
```

- Hiển thị danh sách phòng phòng:

```
Thuc hien chuc nang khac vui long chon [0-13] hoac chá»n 13 de hien MENU
Vui long nhap [0-13]: 3
--HIEN DANH SACH PHONG--
-------------------------------------------------------
ID    Ten         Tang      Loai          Kich co
-------------------------------------------------------
401   Phong401    4         Phong hoc     32
402   Phong402    4         Phong hoc     32
403   Phong403    4         Phong lab     10
404   Phong401    4         Hoi truong    89
301   Phong301    3         Phong hoc     32
302   Phong302    3         Phong hoc     32
303   Phong303    3         Phong lab     14
304   Phong304    3         Hoi truong    100
201   Phong201    2         Phong hoc     32
202   Phong202    2         Phong hoc     32
203   Phong203    2         Phong lab     11
203   Phong204    2         Hoi truong    98
203   Phong204    2         Hoi truong    98
-------------------------------------------------------
```

- Hiển thị danh sách sắp xếp theo kích cỡ:

```
Thuc hien chuc nang khac vui long chon [0-13] hoac chá»n 13 de hien MENU
Vui long nhap [0-13]: 4
--HIEN DANH SACH PHONG SAP XEP THEO KICH CO--
-----------------------------------------------------
ID    Ten         Tang      Loai          Kich co
-----------------------------------------------------
403   Phong403    4         Phong lab     10
203   Phong203    2         Phong lab     11
303   Phong303    3         Phong lab     14
401   Phong401    4         Phong hoc     32
402   Phong402    4         Phong hoc     32
301   Phong301    3         Phong hoc     32
302   Phong302    3         Phong hoc     32
201   Phong201    2         Phong hoc     32
202   Phong202    2         Phong hoc     32
404   Phong401    4         Hoi truong    89
203   Phong204    2         Hoi truong    98
203   Phong204    2         Hoi truong    98
304   Phong304    3         Hoi truong    100
-----------------------------------------------------
```

- Hiển thị danh sách sắp xếp theo tên:

```
Thuc hien chuc nang khac vui long chon [0-13] hoac chá»n 13 de hien MENU
Vui long nhap [0-13]: 5
--HIEN DANH SACH PHONG SAP XEP THEO TEN--
-----------------------------------------------------
ID    Ten         Tang      Loai          Kich co
-----------------------------------------------------
201   Phong201    2         Phong hoc     32
202   Phong202    2         Phong hoc     32
203   Phong203    2         Phong lab     11
203   Phong204    2         Hoi truong    98
203   Phong204    2         Hoi truong    98
301   Phong301    3         Phong hoc     32
302   Phong302    3         Phong hoc     32
303   Phong303    3         Phong lab     14
304   Phong304    3         Hoi truong    100
401   Phong401    4         Phong hoc     32
404   Phong401    4         Hoi truong    89
402   Phong402    4         Phong hoc     32
403   Phong403    4         Phong lab     10
-----------------------------------------------------
```

- Hiển thị danh sach sắp xếp theo tầng:

```
Thuc hien chuc nang khac vui long chon [0-13] hoac chá»n 13 de hien MENU
Vui long nhap [0-13]: 6
--HIEN DANH SACH PHONG SAP XEP THEO TANG--
-------------------------------------------------------
ID    Ten         Tang      Loai          Kich co
-------------------------------------------------------
201   Phong201    2         Phong hoc     32
202   Phong202    2         Phong hoc     32
203   Phong203    2         Phong lab     11
203   Phong204    2         Hoi truong    98
203   Phong204    2         Hoi truong    98
301   Phong301    3         Phong hoc     32
302   Phong302    3         Phong hoc     32
303   Phong303    3         Phong lab     14
304   Phong304    3         Hoi truong    100
401   Phong401    4         Phong hoc     32
402   Phong402    4         Phong hoc     32
403   Phong403    4         Phong lab     10
404   Phong401    4         Hoi truong    89
-------------------------------------------------------
```

- Tìm phòng theo tầng:

```
Thuc hien chuc nang khac vui long chon [0-13] hoac chá»n 13 de hien MENU
Vui long nhap [0-13]: 7
--HIEN DANH SACH THEO TANG--
Nhap tang: 4
-------------------------------------------------------
ID    Ten         Tang      Loai          Kich co
-------------------------------------------------------
401   Phong401    4         Phong hoc     32
402   Phong402    4         Phong hoc     32
403   Phong403    4         Phong lab     10
404   Phong401    4         Hoi truong    89
-------------------------------------------------------
```

- Tìm phòng theo mã phòng:

```
Thuc hien chuc nang khac vui long chon [0-13] hoac chá»n 13 de hien MENU
Vui long nhap [0-13]: 8
--TIM PHONG CO THEO MA PHONG--
Nhap ma phong: 401
-----------------------------------------------------------
ID    Ten         Tang      Loai           Kich co
-----------------------------------------------------------
401   Phong401    4         Phong hoc      32
-----------------------------------------------------------
```

- Cập nhật thông tin phòng:

```
Thuc hien chuc nang khac vui long chon [0-13] hoac chá»n 13 de hien MENU
Vui long nhap [0-13]: 9
--CAP NHAT THONG TIN PHONG--
Nhap ma phong can cap nhat: 401
Moi nhap ID: 401
Moi nhap ten: Phong401Fix
Moi nhap tang: 4
Chon loai phong:
   1. Phong hoc
   2. Phong thi nghiem
   3. Phong hoi thao
Nhap (1/2/3):
1
Moi nhap suc chua phong: 20
Cap nhat phong thanh cong!
```

- Xóa tất cả các phòng:

```
Thuc hien chuc nang khac vui long chon [0-13] hoac chá»n 13 de hien MENU
Vui long nhap [0-13]: 10
--XOA TAT CA CAC PHONG--
Xoa tat ca cac phong thanh cong!
```

- Tìm phòng có kích cỡ lớn hơn hoặc bằng:

```
Thuc hien chuc nang khac vui long chon [0-13] hoac chá»n 13 de hien MENU
Vui long nhap [0-13]: 11
--TIM PHONG CO KICH CO LON HON HOAC BANG--
Nhap kich co: 60
-----------------------------------------------------
ID    Ten         Tang      Loai          Kich co
-----------------------------------------------------
404   Phong401    4         Hoi truong    89
304   Phong304    3         Hoi truong    100
203   Phong204    2         Hoi truong    98
203   Phong204    2         Hoi truong    98
-----------------------------------------------------
```

- Tìm và in danh sách theo loại phòng:

```
Thuc hien chuc nang khac vui long chon [0-13] hoac chá»n 13 de hien MENU
Vui long nhap [0-13]: 12
--TIM VA IN DANH SACH THEO LOAI PHONG--
Chon loai phong:
   1. Phong hoc
   2. Phong thi nghiem
   3. Phong hoi thao
Nhap (1/2/3):
1
-----------------------------------------------------
ID    Ten         Tang      Loai          Kich co
-----------------------------------------------------
401   Phong401Fix 4         Phong hoc     20
402   Phong402    4         Phong hoc     32
301   Phong301    3         Phong hoc     32
302   Phong302    3         Phong hoc     32
201   Phong201    2         Phong hoc     32
202   Phong202    2         Phong hoc     32
-----------------------------------------------------
```

- **Kết quả phần B:**

- Sắp xếp Doubly Linked List:

```
PS G:\DSA\BTL_231220789\problem_b> g++ -std=c++11 Cau1_Test_Sort.cpp -o a
PS G:\DSA\BTL_231220789\problem_b> ./a.exe
Moi nhap so phan tu: 5
Moi nhap phan tu thu 1: 2
Moi nhap phan tu thu 2: 6
Moi nhap phan tu thu 3: 4
Moi nhap phan tu thu 4: 2
Moi nhap phan tu thu 5: 1
Danh sach vua nhap: 2 6 4 2 1
Danh sach sau khi sap xep tang dan: 6 4 2 2 1
Danh sach sau khi sap xep giam dan: 1 2 2 4 6
PS G:\DSA\BTL_231220789\problem_b>
```

- Bài toán tính tuổi con cháu cụ Ted:

```
PS G:\DSA\BTL_231220789\problem_b> ./a.exe
1
1
Ted Bill 25
DATASET 1
Bill 75
PS G:\DSA\BTL_231220789\problem_b>
```

```
PS G:\DSA\BTL_231220789\problem_b> ./a.exe
1
4
Ray James 40
James Beelzebub 17
Ray Mark 75
Ted Ray 20
DATASET 1
Ray 80
James 40
Beelzebub 23
Mark 5
PS G:\DSA\BTL_231220789\problem_b>
```

VI. KẾT LUẬN

1. Đánh giá mức độ hoàn thành

- Đã hoàn thành đúng yêu cầu đề bài.
- Áp dụng tốt các kiến thức đã học trong môn Cấu trúc dữ liệu và giải thuật.
- Phân chia chương trình rõ ràng dễ đọc nâng cấp.

2. Bài học rút ra

- Đây là môn học mang tính khởi đầu mang lại cho em kiến thức để giải quyết các vấn đề thực tế một cách tối ưu, hiệu suất cao nhất.
- Việc áp dụng kiến thức vào thực tế giúp củng cố và hiểu sâu hơn về các lý thuyết, đồng thời khám phá những vấn đề mới.
- Cần có tư duy để thiết kế, chia nhỏ chương trình.

3. Khó khăn với môn học

- Cần hiểu bản chất từng cấu trúc dữ liệu, thuật toán.
- Nắm vững ngôn ngữ lập trình cùng với kiến thức về lập trình hướng đối tượng.

VII. LỜI CẢM ƠN

Em xin gửi lời cảm ơn chân thành đến thầy vì đã tận tâm giảng dạy và hướng dẫn em trong suốt quá trình học môn Cấu trúc Dữ liệu và Giải thuật. Những kiến thức quý báu mà thầy truyền đạt không chỉ giúp em hiểu rõ hơn về các cấu trúc dữ liệu và thuật toán, mà còn giúp em phát triển tư duy logic, khả năng giải quyết vấn đề trong lập trình.

Ngoài ra việc làm bài tập lớn mà thầy đã giao cho đem lại cho em nhiều kĩ năng mới, thử thách mới giúp cho em phát triển một cách toàn diện hơn không chĩ mỗi code mà còn báo cáo, chia sẻ vấn đề.

Cuối cùng, em xin cảm ơn các tài liệu tham khảo và nguồn tài nguyên mà em đã tìm thấy trong quá trình nghiên cứu, giúp em mở rộng kiến thức và áp dụng vào bài tập.

Em xin trân trọng cảm ơn!

.

VIII. TÀI LIỆU THAM KHẢO

- https://www.geeksforgeeks.org/doubly-linked-list/
- Slide bài giảng môn Cấu trúc dữ liệu và giải thuật Khoa Công nghệ thông tin trường Đại học Giao Thông Vận Tải.