1. **Báo cáo tiến độ:**

* **Tuần 1:** Từ ngày 13/12/2019 đến ngày 22/12/2019

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **STT** | **Công việc** | **Thực hiện** | **Nhận xét** | **Chữ ký của GVHD** |
|  | Chọn đề tài, viết đề cương chi tiết và kế hoạch thực hiện. | **X** |  |  |
|  | Thiết kế các lớp đối tượng, cài đặt các phương thức nhập, xuất dữ liệu. | **X** |
|  | Cài đặt phương thức tìm kiếm tuần tự. | **X** |
|  | Cài đặt các phương thức Bubble Sort, Selection Sort. | **X** |
|  | Viết báo cáo tiến độ thực hiện tuần 1. | **X** |

**Nội dung thực hiện:**

* **Thiết kế các lớp đối tượng:**

Các đối tượng chính trong chương trình bao gồm: nút (node) và danh sách(list).

**Đối tượng node(nút):**

Các thuộc tính:

- int value: Giá trị phần tử.

- node \*next: con trỏ chứa địa chỉ của nút tiếp theo.

Các phương thức:

- void setNode(int i): Gán giá trị số nguyên i cho nút.

**Đối tượng list(danh sách):**

Các thuộc tính:

- node \*first: con trỏ đến địa chỉ nút đầu của danh sách.

- node \*last: con trỏ đến địa chỉ nút cuối của danh sách.

Các phương thức:

- void linkNode(node \*&p): gắn một nút vào cuối danh sách

- void swapData(node \*&p1, node \*&p2): đảo giá trị hai nút.

- void keyList(): Nhập giá trị các nút từ bàn phím, dừng khi nhập giá trị rỗng.

- void randList(): Lấy ngẫu nhiên giá trị các nút (0-999).

- void fileList(): Lấy giá trị các nút từ file text “list\_input.txt”.

- void bubbleSort(): Sắp xếp danh sách theo thuật toán bubble sort.

- void selectionSort(): Sắp xếp danh sách theo thuật toán selection sort.

**-** void insertionSort(): Sắp xếp danh sách theo thuật toán insertion sort.

**-** void quickSort(): Sắp xếp danh sách theo thuật toán quick sort.

**-** void mergeSort(): Sắp xếp danh sách theo thuật toán merge sort.

+ void createList(): Khởi tạo và chọn phương pháp nhập dữ liệu cho danh sách.

+ void getList(): Xuất giá trị các nút trong danh sách.

+ void seach(): Tìm kiếm một nút trong danh sách.

+ void sortList(): Chọn phương pháp sắp xếp danh sách và chọn chiều sắp xếp (tăng dần/giảm dần).

* **Cài đặt các phương thức nhập, xuất dữ liệu:**

**void keyList()**: Nhập giá trị nút từ bàn phím, dừng khi nhập giá trị rỗng.

Việc nhập dữ liệu số nguyên trong C++ không chấp nhận giá trị rỗng, để dừng việc nhập dữ liệu khi không có độ dài danh sách xác định, đề tài sử dụng hàm getline() trong thư viện string để nhận dữ liệu kiểu chuỗi, nhận biết kí tự rỗng và dùng lớp đối tượng dựng sẵn stringstream với toán tử >> trong thư viện sstream để chuyển đổi giá trị nhập thành kiểu số nguyên. Phương pháp này cũng giúp nhận biết các trường hợp dữ liệu nhập vào không phải là số.

**void randList()**: Lấy ngẫu nhiên giá trị các nút (0-999).

Để lấy giá trị ngẫu nhiên cho các nút trong danh sách (độ dài danh sách được nhập từ bàn phím), đề tài sử dụng hàm srand(time(NULL)) với hàm time() trong thư viện time.h để khởi tạo các giá trị ngẫu nhiên (giá trị NULL để khởi tạo tại thời điểm thực hiện hàm nhằm cho ra các giá trị ngẫu nhiên khác nhau mỗi lần thực hiện chương trình), và hàm rand() để sinh các giá trị ngẫu nhiên và giới hạn các giá trị được sinh ra trong mội khoảng nhất định.

**void fileList()**: Lấy giá trị các nút từ file text

Để lấy giá trị các nút từ file, đề tài sử dụng biến kiểu FILE và các hàm đọc, ghi file trong thư viện stdio.h. Chương trình sử dụng file text mặc định có tên “list\_input.txt” được đặt trong cùng thư mục. File text bao gồm các số nguyên cách nhau bằng khoảng trống.

**void getList()**: Xuất giá trị các nút trong danh sách

Chương trình in giá trị các nút trong danh sách ra màn hình và ghi ra file có tên “results.txt” trong cùng thư mục (file này được tạo ra nếu không có sẵn).

* **Cài đặt phương thức tìm kiếm tuần tự:**

**void search()**: tìm kiếm một giá trị trong danh sách bằng cách duyệt lần lượt các nút trong danh sách, in giá trị nút tìm được ra màn hình và ghi kết quả vào file “results.txt”.

* **Cài đặt các phương thức sắp xếp bubble sort và selection sort:**

**bubbleSort()**: Dùng hai vòng lặp lồng nhau để duyệt danh sách, so sánh các nút với nhau và gọi phương thức swapData() để đảo giá trị hai nút nếu không đúng thứ tự.

**selectionSort()**: tương tự như bubble sort nhưng sử dụng con trỏ flag để đánh dấu vị trí bắt đầu duyệt danh sách và con trỏ min để đánh dấu nút có giá trị nhỏ nhất. Nếu giá trị nút min khác giá trị nút flag, gọi phương thức swapData() để đảo giá trị hai nút này. Lặp lại đến khi nút flag nằm cuối danh sách.

* **Tuần 2:** Từ ngày 23/12/2019 đến ngày 29/12/2019

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **STT** | **Công việc** | **Thực hiện** | **Nhận xét** | **Chữ ký của GVHD** |
|  | Tìm hiểu và cài đặt thuật toán Insertion Sort. | **X** |  |  |
|  | Tìm hiểu và cài đặt thuật toán Quick Sort. | **X** |
|  | Tìm hiểu và cài đặt thuật toán Merge Sort. | **X** |
|  | Viết báo cáo tiến độ thực hiện tuần 2. | **X** |

**Nội dung thực hiện:**

* + **Cài đặt các phương thức sắp xếp Insertion Sort, Quick Sort:**

**insertionSort():** Tạo một danh sách liên kết trống, lần lượt duyệt các phần tử của danh sách hiện có và chèn vào danh sách mới theo thứ tự bằng cách: tìm nút trong danh sách mới có giá trị nút sau nó lớn/nhỏ hơn giá trị nút đanh xét, chèn nút đang xét vào sau nút tìm được, hoặc nếu nút đang xét có giá trị lớn/nhỏ hơn giá trị nút đầu danh sách mới hay danh sách mới còn trống, chèn nút đang xét vào đầu danh sách mới.

**quickSort():** Chia danh sách thành 2 phân đoạn: Phân đoạn trước gồm các nút có giá trị nhỏ/lớn hơn nút nằm giữa danh sách. Lặp lại bước này cho từng phân đoạn đến khi phân đoạn còn ít hơn hai nút.

Để chia danh sách thành 2 phân đoạn, thay vì tìm nút nằm giữa danh sách và duyệt từ hai đầu danh sách, đề tài chọn nút cuối danh sách làm nút đánh dấu và duyệt từ đầu danh sách. Nếu đầu phân đoạn chưa có và nút đang xét có giá trị lớn/nhỏ hơn giá trị nút đánh dấu thì đặt nút này làm đầu phân đoạn, nếu không thì đặt nút này ra sau nút đánh dấu rồi tiếp tục duyệt đến nút được đánh dấu.

**mergeSort():** Chia danh sách thành hai nửa và lặp lại quá trình sắp xếp cho các đoạn danh sách, sau đó trộn các đoạn danh sách theo thứ tự.

Để trộn theo thứ tự, so sánh từng nút của hai đoạn, nút nào có giá trị nhỏ/lớn hơn thì đặt nút đó vào danh sách, tiếp tục đến khi trộn hết các nút.