|  |  |
| --- | --- |
| BỘ GIÁO DỤC VÀ ĐÀO TẠO  **TRƯỜNG ĐẠI HỌC PHENIKAA**  ĐỀ SỐ: 2  *Đề thi gồm có 5 câu; 3 trang*  *Đề thi* ***không được*** *sử dụng tài liệu* | **ĐỀ THI KẾT THÚC HỌC PHẦN**  *(Đối với môn thi tự luận)*  **Học phần:** Cấu trúc dữ liệu và thuật toán  **Mã học phần:** CSE703006  **Ngày thi:** **Giờ thi**:  Thời gian làm bài: 90 phút  *(Không kể thời gian giao đề)* |

**Họ và tên sinh viên**: ………………………………. **Số báo danh**:……………………

**Câu 1 (2 điểm - CĐR 1.1)**

Cho khai báo kiểu của một nút trong danh sách liên kết như sau:

typedef struct \_listnode {

int num;

struct \_listnode \*next;

} ListNode;

Cho khai báo của các hàm thực hiện các thao tác trên danh sách liên kết như sau:

ListNode\* findNode(ListNode \*head, int i);

Trả về địa chỉ của node thứ i trong danh sách liên kết trỏ bởi head. Node đầu tiên của danh sách liên kết là node thứ 0.

void insertNode(ListNode \*\*pHead, int index, int value);

Chèn một nút có giá trị value vào vị trí index trong danh sách liên kết trỏ bởi \*pHead.

void removeNode(ListNode \*\*ptrHead, int index);

Xóa nút ở vị trí index trong danh sách liên kết trỏ bởi \*pHead.

a) Hoàn thiện hàm findNode()

ListNode \*findNode(ListNode \*head, int i) {

ListNode \*cur = head;

if ((head == NULL) || (i < 0)) {

printf("Danh sach lien ket rong hoac phan tu tim kiem khong ton tai");

return NULL;

}

while (i > 0) {

cur = **cur->next** ;

**i= i-1**

if (cur == NULL) {

printf("Phan tu tim kiem khong ton tai");

return NULL;

}

}

return cur;

}

b) Hoàn thiện hàm insertNode()

void insertNode(ListNode \*\*pHead, int index, int value){

ListNode \*cur, \*newNode;

if (\*pHead == NULL || index == 0){

newNode = malloc(sizeof(ListNode));

newNode->num = value;

newNode->next = \*pHead ;

\*pHead = newNode;

}

else if ((cur = findNode(\*pHead, **index- 1** )) != NULL){

newNode = malloc(sizeof(ListNode));

newNode->num = value;

newNode->next = **cur->next** ;

cur->next = newNode ;

} else printf("Khong the chen phan tu moi tai chi so %d!\n", index);

}

**Câu 2 (2 điểm - CĐR 2.1, 2.2, 2.3, 2.4 có trọng số ngang nhau)**

a) (1 điểm) Vẽ cây nhị phân tìm kiếm (BST) tạo thành bằng việc chèn lần lượt các phần tử của dãy khóa sau đây (bắt đầu từ cây rỗng): 9, 7, 15, 21, 3, 24, 11, 20, 13.

b) (1 điểm) Vẽ cây nhị phân tìm kiếm (BST) tạo thành sau khi loại bỏ nút có khóa bằng 9 ra khỏi cây thu được ở câu a).

**Câu 3 (2 điểm - CĐR 2.1, 2.2, 2.3, 2.4 có trọng số ngang nhau)**

Cho mảng k: k[1], k[2], ..., k[n] chứa dãy n số nguyên.

Cho khai báo của hàm sắp xếp dãy n số theo thứ tự tăng dần dùng giải thuật sắp xếp nổi bọt như sau:

void bubble\_sort(int k[], int n);

a) (1 điểm) Hoàn thiện mã chương trình của hàm bubble\_sort()

void bubble\_sort(int k[], int n) {

int i, j, x;

for (i = 1 ; i < n; i++)

for (j = \_\_\_ ; j \_\_\_ i; j--)

if (k[j] \_\_\_ k[j-1]){

x = k[j];

k[j] = k[j-1];

k[j-1] = x;

}

}

b) (1 điểm) Cho biết kết quả gọi bubble\_sort(b, 7) khi thực hiện xong vòng lặp i = 1 và i = 2 với mảng b khai báo như sau

int b[7] = {0, 99, 66, 22, 55, 11, 88, 77 };

**Câu 4. (3 điểm - CĐR 2.1, 2.2, 2.3, 2.4 có trọng số ngang nhau)**

Cho mảng K: K[0], K[1], ..., K[n] chứa dãy n+1 số nguyên dương.

- Cho khai báo của hàm sắp xếp dãy số nguyên theo thứ tự tăng dần dùng giải thuật sắp xếp MergeSort. h là chỉ số của phần tử đầu, k là chỉ số phần tử cuối của dãy cần sắp xếp.

void MergeSort(int b[], int h, int k);

- Cho khai báo hàm hòa nhập 2 dãy. Dãy thứ nhất b có chỉ số phần tử đầu là h, chỉ số phần tử cuối là t. Dãy thứ hai b có chỉ số phần tử đầu là t+1 và chỉ số phần tử cuối là k. Kết quả dãy sau khi hòa nhập là dãy b có chỉ số từ h tới k.

void Merge(int b[], int h, int t, int k);

a) (1 điểm) Hoàn thiện mã chương trình của hàm Merge()

void Merge(int b[], int h, int t, int k) {

int c[t-h+1];

int i = 0, u = h, v = t+1;

for (i = 0; i <= t-h ; i++)

c[i] = b[ h + i];

i = 0;

while (i <= \_\_\_ ) {

if (v <= \_\_\_ && b[v] < c[i]) {

b[u] = \_\_\_ ;

u++;

v = \_\_\_;

} else {

b[u] = \_\_\_ ;

u++;

i = \_\_\_ ;

}

}

}

b) (1 điểm) Hoàn thiện mã chương trình của hàm MergeSort()

void MergeSort(int b[], int h, int k) {

if (k - h < 1) return;

int t = (h+k) / 2;

MergeSort(b, \_\_\_ , \_\_\_ );

MergeSort(b, \_\_\_ , \_\_\_ );

Merge(b, \_\_\_ , \_\_\_, \_\_\_ );

}

c) (1 điểm) Gọi Merge(b,0,1,4) với dãy int b[5] = {11, 22, 10, 55}. Viết kết quả mảng b và mảng c trước khi vào vòng lặp while. Viết kết quả mảng b sau mỗi lần lặp của vòng while.

**Câu 5. (1 điểm – CĐR 2.1, 2.2, 2.3, 2.4 có trọng số ngang nhau)**

Một bảng băm độ dài 7 sử dụng cơ chế địa chỉ mở và dò tuyến tính với hàm băm h(k) = k mod 7 được sử dụng để chèn các khóa 44, 45, 79, 55, 91, 18, 63 vào các ô của bảng. Vị trí của khóa 18 là bao nhiêu. Giải thích.

a. 3 b. 4 c. 6 d. 5

TRƯỞNG BỘ MÔN/KHOA GIẢNG VIÊN RA ĐỀ