BỘ XÂY DỰNG
TRƯỜNG ĐHXD MIỀN TÂY

ĐÁP ÁN ĐỀ THI CHÍNH THỨC

ĐÁP ÁN – THANG ĐIỂM
KỲ THI KTHP HỌC KỲ II NĂM HỌC 2024-2025
Trình độ: ĐẠI HỌC; Ngày thi: 19/5/2025
Môn: CẤU TRÚC DỮ LIỆU
(Đáp án - thang điểm gồm 06 trang)

Lưu ý: Chương trình thực thi in ra màn hình kết quả đúng sinh viên sẽ được nhận tối đa số điểm câu đó. Nếu chương trình thực thi lỗi chấm theo thang điểm code bên dưới.

Câu	Phần	Nội dung	Thang điểm

1	**a**	// Khai báo cây tìm kiếm nhị phân
		#include <stdio.h>
		#include<malloc.h>
		typedef int DataType;
		struct Node{
		DataType Data;
		Node* left;
		Node* right;
		};
		typedef Node* Tree;
			0.5

		// Tạo cây rỗng
		void MakeNull_Tree(Tree *T) {
		(*T)=NULL; }
			0.5

		//Kiểm tra cây rỗng
		int EmptyTree(Tree T) {
		return T==NULL;
		}	

		// Xác định con trái
		Tree LeftChild(Tree T) {
		if(T != NULL)
		return T->left;
		else return NULL; }
			0.5

		// Xác định con phải
		Tree RightChild(Tree T) {
		if(T != NULL)
		return T->right;
		else return NULL;
		}	

		//Kiểm tra nút lá int isLeaf(Tree T){	**0.5**

Câu	Phần	Nội dung	Thang điểm

		if((T != NULL) && (T->left == NULL) && (T->right == NULL))
		return 1;
		else return 0;
		}
		//Tạo cây mới từ 2 cây có sẵn
		Tree Create(DataType V,Tree L,Tree R) {
		Tree N; // Khai bao 1 cay moi
		N = (Node*)malloc(sizeof(Node));
		N->Data = V;
		N->left = L;
		N->right = R;
		return N;
		}
			0.5

	b	//Xác định số nút của cây
		int NumberNode(Tree T){
		if(EmptyTree(T))
		return 0;
		else return 1 + NumberNode(LeftChild(T))+NumberNode(RightChild(T));
		}
			0.5

	c	// Chiều cao của cây
		int max(int value1, int value2) {
		return ((value1 > value2) ? value1 : value2); }
		int TreeHeight(Tree T) {
		int height;
		if (EmptyTree(T)||isLeaf(T))
		return 0;
		else return height=max(TreeHeight(LeftChild(T)),TreeHeight(RightChild(T))+1; }
			1.0

	d	// Thêm nút có khóa x vào cây
		void Them (DataType x, Tree *T){
		if((*T) == NULL) {
		(*T)= (Node*)malloc(sizeof(Node));
		(*T)->Data = x ;
			0.5

Câu	Phần	Nội dung	Thang điểm

		(*T)->left = NULL;
		(*T)->right = NULL; }
		else if((*T)->Data == x)
		printf("Da ton tai khoa X");
		else if((*T)-> Data > x)
		Them(x,&(*T)->left);
		else Them(x,&(*T)->right);
		}	

	e	//Duyet trung tu
		void TrungTu(Tree T) {
		if(!EmptyTree(T)) {
		TrungTu(LeftChild(T));
		printf("%d ",T->Data);
		TrungTu(RightChild(T));
		}
		}
			0.5

	f	main(){
		Tree T,T1,T2;
		T1=
		Create(12,Create(4,NULL,NULL),Create(20,NULL,NULL));
		T2=
		Create(40,Create(34,Create(30,NULL,NULL),NULL),Create(50,
		NULL,NULL));
		T = Create(27,T1,T2);
		printf("\nDanh sach duyet trung tu: ");
		printf ("So nut hien co tren cay la %d ", NumberNode(T));
		printf("\nChieu cao cua cay la: %d",TreeHeight(T));
		printf("\n\nNhap gia tri muon them vao cay: ");
		scanf("%d", &x);
		Them(x, &T);
		printf("\nCay sau khi them gia tri %d (duyet trung tu):\n",x);
		TrungTu(T);
		}
			1.0

2	**a**	// Khai báo danh sách #include<stdio.h> #define MaxLength 30	**1.0**

Câu	Phần	Nội dung	Thang điểm

		typedef int ElementType;
		typedef int Position;
		typedef struct {
		ElementType Elements[MaxLength];
		Position Last;
		}List;
		// Khởi tạo danh sách rỗng
		void MakeNull_List(List *L){
		L->Last = 0;
		}
		// Kiểm tra danh sách rỗng
		int Empty_List(List L){
		return (L.Last == 0); }	

	b	// Hàm trả về vị trí phần tử đầu tiên trong danh sách
		Position FirstList(List L){
		return 1;
		}
		// Hàm trả về vị trí sau phần tử cuối cùng trong danh sách
		Position EndList(List L){
		return L.Last+1; }
			0.5

	c	// Hàm trả về vị trí phần tử kế tiếp phần tử P trong danh sách
		Position Next(Position P, List L){
		return P+1;
		}
		// Hàm trả về nội dung phần tử trong danh sách
		ElementType Retrieve(Position P, List L){
		return L.Elements[P-1]; }
			0.5

	d	// Xóa phần tử tại vị trí P trong danh sách
		void Insert_List(ElementType X, Position P, List *L){
		int i=0;
		if(L->Last == MaxLength)
		printf("\nDanh sach day !");
		else if ((P<1) || (P>L->Last+1))
		printf("\nVi tri khong hop le !!!");
		else { for(i=L->Last ;i>=P ; i--)
		L->Elements[i] = L->Elements[i-1];
			0.5

Câu	Phần	Nội dung	Thang điểm
		L->Last++; L->Elements[P-1] = X; } }	

2	e	// Nhập danh sách từ bàn phím.
		void Read_List(List *L) {
		int i, N;
		ElementType X;
		MakeNull_List(L);
		printf("\nNhap vao so phan tu trong danh sach: ");
		scanf("%d", &N);
		if (N > MaxLength) {
		printf("\nSo phan tu vuot qua gioi han cho phep\n", MaxLength);
		N = MaxLength;
		}
		for (i = 0; i < N; i++) {
		printf("\nPhan tu thu %d la: ", i + 1);
		scanf("%d", &X);
		L->Elements[i] = X;
		}
		L->Last = N;
		}
			0.5

	f	//Hiển thị danh sách ra màn hình
		void Print_List(List L){
		Position P;
		P = FirstList(L);
		printf("\nBat dau in danh sach ");
		while(P != EndList(L)){
		printf("\n%d",Retrieve(P,L));
		P = Next(P,L); }
		printf("\nKet thuc in danh sach");
		}
			0.5

	g	// Thực thi hàm xóa tại một vị trí trong danh sách và hiển thị danh
		// sách ra màn hình
		main(){
		List L;
			0.5

Câu	Phần	Nội dung	Thang điểm
		int pos; MakeNull_List(&L); Read_List(&L); printf("\nDanh sach vua nhap la:"); Print_List(L); printf("\nNhap vi tri can xoa: "); scanf("%d", &pos); Delete_List(pos, &L); printf("\nDanh sach sau khi xoa la:"); Print_List(L);}	
Tổng điểm			**10.0 đ**