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 2022-2023
Trình độ: ĐẠI HỌC; Ngày thi: 12/06/2023
Môn: CẤU TRÚC DỮ LIỆU
(Đáp án - thang điểm gồm 07 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	`#include <stdio.h>`
		`#include <stdlib.h>`
		`#define MaxLength 50`
		`typedef int ElementType;`
		`typedef int Position;`
		`typedef struct {`
		`  ElementType Elements[MaxLength];`
		`  Position Last;`
		`} List;`
		
		`List L;`
		
		`void MakeNullList(List *L) {`
		`  L->Last=0;`
		`}`
		
		`int EmptyList(List L) {`
		`  return (L.Last==0);`
		`}`
		`ElementType RetrievePosition P, List L) {`
		`  return L.Elements[P-1];`
		`}`
		
		`Position First(List L) {`
		`  return 1;`
		`}`
		
		`Position EndList(List L) {`
		`  return L.Last+1;`
		`}`
		
		`Position Next(Position P, List L) {`
		`  return P+1;`
		`}`
			0.5

		`void InsertList(ElementType X,Position P, List *L)`	0.5

```
{
     if (L->Last==MaxLength)
          printf("Danh sach day");
     else if ((P<1) || (P>L->Last+1))
         printf("Vi tri khong hop le");
          else {
               Position Q;
          for(Q=(L->Last-1)+1;Q>=P;Q--)
          L->Elements[Q]=L->Elements[Q-1];
               L->Elements[P-1]=X;
               L->Last++;
          }
}    else if ((P<1) || (P>L->Last+1))
         printf("Vi tri khong hop le");
          else {
               Position Q;
               for(Q=(L->Last-1)+1;Q>=P;Q--)
          L->Elements[Q]=L->Elements[Q-1];
               L->Elements[P-1]=X;
               L->Last++;
          }
}
```

a — **0.5**

```
void PrintList(List L) {
     Position P;
     P = First(L);
     while (P != EndList(L)) {
          printf("%d ",Retrieve(P,L));
          P = Next(P, L);
     }
     printf("\n");
}
void ReadList(List *L) {
     int i,N;
     ElementType X;
     MakeNullList(L);
     printf("So phan tu danh sach N= ");
     scanf("%d",&N);
     for(i=1;i<=N;i++) {
          printf("Phan tu thu %d: ",i);
          scanf("%d",&X);
          InsertList(X,EndList(*L),L);
     }
```

b — **1.0**

```
int count(List L) {
     return L.Last;
}

int count1(List L) {
     int sophtu = 0;
     Position P = First(L);
```

```
     while(P != EndList(L)) {
          sophtu++;
          P = Next(P, L);
     }
     return sophtu;
}

main() {
     List L;
     ElementType X;
     Position P;
     ReadList(&L);
     printf("Danh sach vua nhap: ");
     PrintList(L);
printf("So phan tu trong danh sach: %d\n",
count(L));
     return 0;
}
```

2 **a** **0.5**

```
#include<stdio.h>
#include<conio.h>
#include<cstdlib>
#define Max 100
typedef int ElementType;
typedef struct Node
{
     ElementType Element;
     Node* Next;
};
     typedef Node* Position;
     typedef Position Stack;

void MakeNull_Stack(Stack *Header){
     (*Header)=(Node*)malloc(sizeof(Node));
     (*Header)->Next= NULL;
     }
int Empty_Stack(Stack S){
     return (S->Next == NULL);
}
ElementType Top(Stack S){
     if (! Empty_Stack(S))
     return S->Next->Element;
}
```

a **0.5**

```
void Push(ElementType x, Stack *S)
{
     Position temp;
     temp = (Node*)malloc(sizeof(Node));
     temp->Element = x;
     temp->Next = (*S)->Next;
     (*S)->Next = temp;
```

		}
		void Pop(Stack *S){
		if (! Empty_Stack(*S)){
		Position temp;
		temp = (*S)->Next;
		(*S)->Next = temp->Next;
		free(temp);
		}
		}	

	a	void Read_Stack(Stack *S){ int i,N;
		ElementType x;
		printf("So phan tu danh sach N = ");
		scanf("%d",&N);
		for(i=1; i<=N; i++){
		printf("Phan tu thu %d: ", i);
		fflush(stdin);
		scanf("%d", &x);
		Push(x, S);
		}
		}
		void Print_Stack(Stack *S){
		while(! Empty_Stack(*S)) {
		printf("%d \t", Top(*S));
		Pop(S);
		}
		printf("\n");
		}
			0.5

	b	int main(){
		Stack S;
		ElementType n, temp, Fibo;
		MakeNull_Stack(&S);
		Read_Stack(&S);
		Print_Stack(&S);
		printf("Nhap so thap phan n = ");
		scanf("%d",&n);
		if(n == 0)
		printf("So nhi phan tuong ung: 0");
		else{
		temp = n; MakeNull_Stack(&S);
		while(temp >0){
		Push(temp % 2, &S);
		temp = temp/2;
		}
		printf("So nhi phan tuong ung: ");
		while(! Empty_Stack(S)){
		printf("%d", Top(S)); Pop(&S);
		}
		}
		printf("\n");
		getch();
			1.0

```
    return 0;

}
```

3

```
#include <stdio.h>
#include <stdlib.h>
typedef int KeyType;
typedef struct Node* NodeType;
struct Node {
    KeyType key;
    NodeType left,right;
};

typedef NodeType Tree;

void MakeNullTree(Tree *T) {
    (*T)=NULL;
}
```

0.5

```
int EmptyTree(Tree T) {
    return T==NULL;
}
void InsertNode(KeyType x, Tree *Root) {
    if (*Root == NULL) {

    (*Root)=(NodeType)malloc(sizeof(struct
Node));
        (*Root)->key = x;
        (*Root)->left = NULL;
        (*Root)->right = NULL;
    }
    else if (x < (*Root)->key)
InsertNode(x, &((*Root)->left));
    else if (x > (*Root)->key)
InsertNode(x, &((*Root)->right));
}
```

0.5

```
Tree LeftChild(Tree n) {
    if (n!=NULL)
        return n->left;
    else
        return NULL;
}

Tree RightChild(Tree n){
    if (n!=NULL)
        return n->right;
    else
        return NULL;
}
```

0.5

		int IsLeaf(Tree n) {
		if(n!=NULL)
		
		return(LeftChild(n)==NULL)&&(RightChild(n)==NULL);
		else
		return NULL;
		}	

	a	void PreOrder(Tree T) {
		if(T != NULL) {
		printf("%d ",T->key);
		PreOrder(LeftChild(T));
		PreOrder(RightChild(T));
		}
		}
			0.5

	a	void InOrder(Tree T) {
		if(T != NULL) {
		InOrder(LeftChild(T));
		printf("%d ",T->key);
		InOrder(RightChild(T));
		}
		}
			0.5

	a	void PostOrder(Tree T) {
		if (T!=NULL) {
		PostOrder(LeftChild(T));
		PostOrder(RightChild(T));
		printf("%d ",T->key);
		}
		}
			0.5

	b	int nb_nodes(Tree T) {
		if(EmptyTree(T))
		return 0;
		else
		return 1 + nb_nodes(LeftChild(T)) + nb_nodes(RightChild(T));
		}
			0.5

		int max(int x, int y) {
		int ret = (x>y)?x:y;
		return ret;
		}
		int h(Tree n) {
		if (IsLeaf(n) || (n == NULL))
		return 0;
		else
		return 1+max(h(n->left), h(n->right));
		}
			0.5

		main() {
		Tree tree, node;
		KeyType x;
		int i, n;
		MakeNullTree(&tree);
		printf("So nut nhap vao: ");
		scanf("%d", &n);
		for(i=0; i<n; i++) {
		printf("Nhap vao nut thu %d: ",i+1); scanf("%d", &x);
		InsertNode(x, &tree);
		}
		printf("\n duyet tien tu :");
		PreOrder(tree);
		printf("\n duyet trung tu :");
		InOrder(tree);
		printf("\n duyet hau tu :");
		PostOrder(tree);
		printf("\n so nut cua cay la %d \n", nb_nodes(tree));
		printf("\n chieu cao cua cay la %d\n",h(tree));
		}
			1.0

TỔNG ĐIỂM			**10.0**