Trần Đạt Tín
TNK49
2011345

Câu 2:

```
#include "stdio.h"

typedef int ElementType;

typedef struct node
{
    ElementType Data;
    struct node *Next;
} NodeType;

typedef NodeType* NodePointer;

typedef struct {
    NodePointer Head, Tail;
} LL;

NodePointer TaoNut(ElementType X)
{
    NodePointer np;
    np = new NodeType;
    if (np == NULL)
        printf("Lỗi cấp phát bộ nhớ");
    else
    {
```

```
        np->Data = x;
        np->Next = NULL;
    }
    return np;
}

LL TaoDSRong()
{
    LL L;
    L.Head = L.Tail = NULL;
    return L;
}

bool KiemTraDSLKRong(LL L)
{
    if (L.Head == NULL)
        return true;
    return false;

// a)
void ChenCuoi(LL &L, ElementType X)
{
    NodePointer np;
    np = TaoNut(X);
    if (KiemTraDSLKRong(L))
    {
        L.Head = np;
        L.Tail = np;
    }
    else
```

```
    {
        L.Tail -> Next = np;
        L.Tail = np;
    }
}

LL TaoDS()
{
    LL L;
    L = TaoDSRong();
    do
    {
        ElementType x;
        printf("Enter data: ");
        scanf("%d", &x);
        if (x == -1)
            break;
        ChenCuoi(L, x);
    } while (true);
    return L;
}

int XuatDS(LL L)
{
    if (KiemTraDSLKRong(L) == true)
        return -1;
    NodePointer cur = L.Head;
    while (cur != NULL)
    {
        printf("%d ", cur -> Data);
```

```
        cur = cur -> Next;
    }
    return 1;
}

// b)
void findValue (LL L, ElementType X)
{
    NodePointer cur = L.Head;
    while (cur != NULL)
    {
        if ( cur -> Data > X && cur -> Data % 2 == 0)
        {
            printf ("Tìm được: %d", cur -> Data);
            break;
        }
        cur = cur -> Next;
    }
    if (cur == NULL)
        printf ("Không có giá trị cần tìm trong danh sách");
}

// c)

NodePointer TimGiaTri (LL L, ElementType X)
{
    NodePointer cur = L.Head;
    while (cur != NULL)
    {
        if (cur -> Data > X && cur -> Data % 2 == 0)
```

```
            return cur;
        cur = cur -> Next;
    }
    return NULL;
}

Void mixLinkList (LL& L1, NodePointer M, LL L2)
{
    if (M == L1.Tail)
    {
        M -> Next = L2.Head;
        L1.Tail = L2.Tail;
    }
    else
    {
        L2.Tail -> Next = M -> Next;
        M -> Next = L2.Head;
    }
}

Void ChenDanhSach (LL L1, LL L2, ElementType X)
{
    NodePointer cur = TimGiaTri (L1, X);
    if (cur == NULL)
        cur = L1.Tail;
    mixLinkList (L1, cur, L2);
}
```

```
//d)
void removeNode (LL& L, NodePointer& pre_node,
NodePointer& cur_node)
{
    NodePointer tmp = cur_node;
    if (cur_node == L.Head)
    {
        L.Head = L.Head -> Next;
        cur_node = curnode -> Next;
    }
    else
    {
        cur_node = cur_node -> Next;
        pre_node -> Next = cur_node;
    }
    tmp -> Next = NULL
    delete tmp;
}

LL removeTwoNode (LL L, int X)
{
    if (KiemTraDSLKRong(L))
        return L;
    NodePointer cur_node = L.Head;
    NodePointer pre_node = NULLL;
    int count = 0;
    while (cur_node != NULL && count < 2)
    {
        if (cur_node -> Data < X)
```

```
        {
            removeNode(L, pre_node, cur_node);
            count++;
        }
        else
        {
            pre_node = cur_node;
            cur_node = cur_node->Next;
        }
    }
    return 2;
}
// e)
int main()
{
    LL L1 = TaoDSRong();
    LL L2;
    int x, y;
    int choice = -1;
    while (choice != 0)
    {
        printf("What do you want to do?\n");
        printf("[0] Exit\n");
        printf("[1] Chèn Cuối vào L1\n");
        printf("[2] Tìm trong L1 và xuất ra giá trị chẵn đầu tiên lớn hơn X\n");
        printf("[3] Chèn L2 vào L1 sau một giá trị chẵn đầu tiên lớn hơn X trong L1\n");
        printf("[4] Xoá hết nút đầu tiên có dữ liệu nhỏ hơn X cho trước trong L1\n");
```

```
scanf("%d", &choice);
switch(choice)
{
case 0:
    return 0;
    break;
case 1:
    printf("X = ");
    scanf("%d", &x);
    ChenCuoi(L1, x); break;
Case 2:
    printf("X = ");
    Scanf("%d", &x);
    findValue(L1, x); break;
case 3:
    printf("Nhập dữ Liệu L2");
    L2 = TaoDS();
    printf("X = ");
    Scanf("%d", &x);
    ChenDanhSach(L1, L2, x);
    printf("Before \n");
    XuatDS(L1);
    printf("After \n");
    XuatDS(L1);
    break;
case 4:
    printf("X = ");
    scanf("%d", &x);
    printf("Before \n");
    XuatDS(L1);
```

```
            L1 = removeTwoNode(L1, x);
            printf("After\n");
            XuatDS(L1);
            break;
        default:
            break;
    }
    printf("\n\n");
}
}
```

Câu 1

```
#include <stdio.h>

typedef int array[100];

//a)

void inputArray(array a, int &n)
{
    printf("size array a: ");
    scanf("%d", &n);
    for(int i=0; i<n; i++)
    {
        printf("a[%d] = ", i+1);
        scanf("%d", &a[i]);
    }
}
```

```
Void outputArray(array a, int n)
{
    printf("** Output Array **\n");
    for(int i=0; i<n; i++)
    {
        printf("%d ", a[i]);
    }
    printf("\n");
}

//b)
int findMax(array a, int n)
{
    int res = a[0];
    for(int i=1; i<n; i++)
    {
        if(a[i] > res)
            res = a[i];
    }
    return res;
}

//c)

bool check(int n)
{
    int sum = 0;
    for(int i=1; i<=n/2; i++)
    {
        if(n%i == 0)
```

```
        sum += i;
    }
    if (sum == n) return true;
    return false;
}

Void TimSoHoanChinh(Carray a, int n)
{
    printf("So hoan chinh trong mang A: ");
    for(int i = 0; i<n; i++)
    {
        if (check(a[i]) == true)
        {
            printf("%d ", a[i]);
        }
    }
    printf("\n");
}

// d)
Void swap(int &a, int &b)
{
    int t = a; a = b; b = t;
}

int minIndex(int a[], int i, int n, int& co)
{
    int min_index = i; co = 1;
    for(int j = i+1; j<n; j++)
        if(a[j] < a[min_index] && a[j] %2 == 0)
```

```
            min_index = j ;
        if (a[min_index] % 2 != 0)
            co = -1 ;
        return min_index;
}

Void selectSort (int arr[], int n)
{
    int i, min_index, co;
    for (i=0; i<n-1 ; i++)
    {
        min_index = minIndex (arr, i+1, n, co);
        if ( co == -1)
            break;
        if (arr [min_index], arr [i]);
    }
}

int main()
{
    array a;
    int n;
    inputArray (a, n);
    outputArray (a, n);
    printf ("%d ", findMax (a, n));
    TimSothoomChinh (a, n);
    selectSort (a, n);
    outputArray (a, n);
}
```