DANH SÁCH KỀ (tt)

(1) Sửa bài tập về nhà buổi 7:

a. Xây dựng CT có menu lệnh chọn lựa các xử lý danh sách kề:

```
#define _CRT_SECURE_NO_WARNINGS
#include<stdio.h>
#include<conio.h>
#define SIZEMAX 50
struct LIST {
    int num;
    int nodes[SIZEMAX];
};
void Initialize(LIST *list) {
    list->num = 0;
}
int ListSize(LIST list) {
    return list.num;
}
bool Empty(LIST list)
{
    return list.num == 0 ? true : false;
}
bool Full(LIST list)
{
    return list.num == SIZEMAX ? true : false;
}
void Insert(LIST *list, int pos, int x)
{
    if (pos < 0 || pos > list->num)
            printf("Vi tri %d khong hop le\n", pos);
    else if (Full(*list))
            printf("Danh sach bi day\n");
    else
    {
            for (int i = list->num - 1; i >= pos; i--)
                    list->nodes[i + 1] = list->nodes[i];
            list->nodes[pos] = x;
            list->num++;
    }
}
void Remove(LIST *list, int pos)
{
    if (pos < 0 || pos >= list->num)
            printf("Vi tri %d khong hop le\n", pos);
    else if (Empty(*list))
            printf("Danh sach bi rong\n");
    else
    {
            int x = list->nodes[pos];
            for (int i = pos; i<list->num - 1; i++)
                    list->nodes[i] = list->nodes[i + 1];
```

```
                list->num--;
                printf("Da xoa phan tu co gia tri la %d tai vi tri %d\n",x,pos);
        }
}
void Replace(LIST *list, int pos, int x)
{
        if (pos < 0 || pos >= list->num)
                printf("Vi tri %d khong hop le\n", pos);
        else if (Empty(*list))
                printf("Danh sach bi rong\n");
        else
                list->nodes[pos] = x;
}
void Sort(LIST *list)
{
        for (int i = 0; i < list->num - 1; i++)
                for (int j = i + 1; j < list->num; j++)
                        if (list->nodes[i] > list->nodes[j])
                        {
                                int tmp = list->nodes[i];
                                list->nodes[i] = list->nodes[j];
                                list->nodes[j] = tmp;
                        }
}
int LinearSearch(LIST list, int x)
{
        for (int i = 0; i < list.num; i++)
        {
                if (list.nodes[i] == x)
                        return i;
        }
        return -1;
}
int BinarySearch(LIST list, int x)
{
        int dau = 0;
        int cuoi = list.num - 1;
        int giua;
        while (dau <= cuoi)
        {
                giua = (dau + cuoi) / 2;
                if (x == list.nodes[giua])
                        return giua;
                if (x < list.nodes[giua])
                        cuoi = giua - 1;
                else
                        dau = giua + 1;
        }
        return -1;
}
```

```
void Traverse(LIST list)
{
    if (Empty(list))
            printf("Danh sach rong\n");
    else
    {
            for (int i = 0; i < list.num; i++)
                    printf("%d\t", list.nodes[i]);
            printf("\n");
    }
}
void ClearList(LIST *list)
{
    list->num = 0;
}
void InputList(LIST *list)
{
    do
    {
            printf("Nhap so nut: ");
            scanf("%d", &list->num);
    } while (list->num<1 || list->num>SIZEMAX);
    for (int i = 0; i<list->num; i++)
    {
            printf("Nhap gia tri nut thu %d: ", i);
            scanf("%d", &list->nodes[i]);
    }
}
void HoanVi(int &so1, int &so2)
{
    int tam = so1;
    so1 = so2;
    so2 = tam;
}
void SortHaftList(LIST *list)
{
    int i, j;
    int mid = list->num / 2;
    //Sap xep nua dau tang dan
    for (i = 0; i<mid - 1; i++)
            for (int j = i + 1; j<mid; j++)
                    if (list->nodes[i]>list->nodes[j])
                            HoanVi(list->nodes[i], list->nodes[j]);
    //Sap xep nua sau giam dan
    for (i = mid; i<list->num - 1; i++)
            for (j = i + 1; j<list->num; j++)
                    if (list->nodes[i]<list->nodes[j])
                            HoanVi(list->nodes[i], list->nodes[j]);
}
```

```
void SplitList(LIST list, LIST *list1, LIST *list2)
{
    int pos;
    do
    {
            printf("Nhap vi tri can tach d/s: ");
            scanf("%d",&pos);
    } while (pos<0 || pos>list.num);
    list1->num = 0;
    list2->num = 0;
    //Tach nua d/s list phia truoc vi tri pos dua vao list1
    for (int i = 0; i<pos; i++)
            list1->nodes[list1->num++] = list.nodes[i];
    //Tach nua d/s list phia sau vi tri pos dua vao list2
    for (int j = pos; j<list.num; j++)
            list2->nodes[list2->num++] = list.nodes[j];
    printf("Danh sach 1:\n");
    Traverse(*list1);//duyet d/s list1
    printf("Danh sach 2:\n");
    Traverse(*list2);//duyet d/s list2
}
void MergeList(LIST list1, LIST list2, LIST *list3)
{
    int i = 0, j = 0;
    list3->num = 0;
    while (i<list1.num&&j<list2.num)
            if (list1.nodes[i]<list2.nodes[j])
                    list3->nodes[list3->num++] = list1.nodes[i++];
            else
                    list3->nodes[list3->num++] = list2.nodes[j++];
    while (i<list1.num)
            list3->nodes[list3->num++] = list1.nodes[i++];
    while (j<list2.num)
            list3->nodes[list3->num++] = list2.nodes[j++];
    Traverse(*list3);
}
int Search(LIST list, int x, int pos)
{
    for (int i = pos + 1; i<list.num; i++)
    if (list.nodes[i] == x)
            return i;
    return -1;
}
void Remove1(LIST *list, int pos)
{
    for (int i = pos; i<list->num; i++)
            list->nodes[i] = list->nodes[i + 1];
    list->num--;
}
```

```
void FilterList(LIST *list)
{
    for (int i = 0; i<list->num; i++)
    {
            while (Search(*list, list->nodes[i], i) != -1)
                    Remove1(list, Search(*list, list->nodes[i], i));
    }
    Traverse(*list);
}
void main()
{
    LIST l,l1,l2,l3;
    int chon;
    Initialize(&l);
    do
    {
            printf("\n-------CHUONG TRINH XU LY D/S KE------------------------------------\n");
            printf("1. Khoi dong d/s\n");
            printf("2. Xac dinh so nut trong d/s\n");
            printf("3. Kiem tra d/s co rong khong?\n");
            printf("4. Kiem tra d/s co day khong?\n");
            printf("5. Them 1 nut vao d/s\n");
            printf("6. Xoa 1 nut khoi d/s\n");
            printf("7. Thay nut trong d/s bang 1 nut khac\n");
            printf("8. Sap xep d/s theo thu tu tang dan\n");
            printf("9. Tim kiem tuyen tinh 1 nut trong d/s\n");
            printf("10. Tim kiem nhi phan 1 nut trong d/s\n");
            printf("11. Duyet tat ca cac nut cua d/s\n");
            printf("12. Xoa tat ca cac nut cua d/s\n");
            printf("13. Nhap d/s cac nut\n");
            printf("14. Sap xep d/s nua dau tang dan, nua sau giam dan\n");
            printf("15. Chia d/s tai vi tri pos thanh 2 d/s moi\n");
            printf("16. Tron 2 d/s co thu tu tang dan thanh 1 d/s cung co thu tu tang dan\n");
            printf("17. Loc d/s loai bo cac nut trung\n");
            printf("0. Thoat CT\n");
            printf("------------------------------------------------------------------\n");
            printf("Ban chon: ");
            scanf("%d", &chon);
            switch (chon)
            {
            case 0:
                    printf("Dang thoat CT...");
                    break;
            case 1:
                    Initialize(&l);
                    break;
            case 2:
                    if (ListSize(l) == 0)
                            printf("D/s rong, chua co nut.");
                    else
                            printf("So nut trong d/s la %d", ListSize(l));
                    break;
```

```
        case 3:
            if (Empty(l))
                printf("D/s rong");
            else
                printf("D/s khong rong");
            break;
        case 4:
            if (Full(l))
                printf("D/s day");
            else
                printf("D/s chua day");
            break;
        case 5:
            int pos, x;
            printf("Nhap vi tri nut can them vao d/s: ");
            scanf("%d", &pos);
            printf("Nhap gia tri nut can them vao d/s: ");
            scanf("%d", &x);
            Insert(&l, pos, x);
            printf("D/s ket qua:\n");
            Traverse(l);
            break;
        case 6:
            if (Empty(l))
                printf("D/s rong");
            else
            {
                printf("Nhap vi tri nut muon xoa trong d/s: ");
                scanf("%d", &pos);
                Remove(&l, pos);
                if (Empty(l))
                    printf("D/s rong");
                else
                {
                    printf("D/s sau khi xoa la:\n");
                    Traverse(l);
                }
            }
            break;
        case 7:
            if (Empty(l))
                printf("D/s rong");
            else
            {
                printf("Nhap vi tri nut can thay trong d/s: ");
                scanf("%d", &pos);
                printf("Nhap gia tri nut thay the: ");
                scanf("%d", &x);
                Replace(&l, pos, x);
                printf("D/s sau khi thay the nut la:\n");
                Traverse(l);
            }
```

```
				break;
			case 8:
				if (Empty(l))
					printf("D/s rong");
				else
				{
					Sort(&l);
					printf("D/s sau khi sap xep tang dan la:\n");
					Traverse(l);
				}
				break;
			case 9:
				if (Empty(l))
					printf("D/s rong");
				else
				{
					printf("Nhap gia tri nut can tim tuyen tinh: ");
					scanf("%d", &x);
					if (LinearSearch(l, x) != -1)
						printf("Tim thay nut co gia tri la %d tai vi tri %d trong d/s", x,
LinearSearch(l, x));
					else
						printf("Khong tim thay");
				}
				break;
			case 10:
				if (Empty(l))
					printf("D/s rong");
				else
				{
					Sort(&l);
					printf("Mang sap xep tang dan la:\n");
					Traverse(l);
					printf("Nhap gia tri nut can tim nhi phan: ");
					scanf("%d", &x);
					if (BinarySearch(l, x) != -1)
						printf("Tim thay nut co gia tri la %d tai vi tri %d trong d/s", x,
BinarySearch(l, x));
					else
						printf("Khong tim thay");
				}
				break;
			case 11:
				if (Empty(l))
					printf("D/s rong");
				else
				{
					printf("Cac nut trong d/s la:\n");
					Traverse(l);
				}
				break;
```

```
                case 12:
                        if (Empty(l))
                                printf("D/s rong");
                        else
                        {
                                ClearList(&l);
                                printf("D/s rong");
                        }
                        break;
                case 13:
                        InputList(&l);
                        printf("D/s ket qua:\n");
                        Traverse(l);
                        break;
                case 14:
                        if (Empty(l))
                                printf("D/s rong");
                        else
                        {
                                SortHaftList(&l);
                                printf("D/s ket qua:\n");
                                Traverse(l);
                        }
                        break;
                case 15:
                        if (Empty(l))
                                printf("D/s rong");
                        else
                                SplitList(l, &l1, &l2);
                        break;
                case 16:
                        printf("Nhap d/s 1:\n");
                        InputList(&l1);
                        Sort(&l1);
                        printf("Nhap d/s 2:\n");
                        InputList(&l2);
                        Sort(&l2);
                        MergeList(l1, l2, &l3);
                        break;
                case 17:
                        if (Empty(l))
                                printf("D/s rong");
                        else
                                FilterList(&l);
                        break;
                default: printf("Ban chon sai. Moi chon lai");
                }
        } while (chon != 0);
    }
```

b. Viết chương trình có cài đặt danh sách kề để quản lý danh sách viên (có cấu trúc gồm: mã SV kiểu số, họ tên SV kiểu chuỗi). CT có các chức năng như sau:

- ✓ *Thêm 1 SV vào danh sách*
- ✓ *Xóa 1 SV khỏi danh sách*
- ✓ *Hiệu chỉnh SV*
- ✓ *Xem danh sách SV*
- ✓ *Sắp xếp danh sách theo mã SV*
- ✓ *Tìm kiếm SV theo mã SV*
- ✓ *Xóa toàn bộ danh sách*

```
#define _CRT_SECURE_NO_WARNINGS
#include<stdio.h>
#include<conio.h>
#include<string.h>
#define SIZEMAX 50
struct SINHVIEN
{
	char ms[5];
	char ten[15];
};
struct LIST {
	int num;
	SINHVIEN nodes[SIZEMAX];
};
void NhapSV(SINHVIEN *sv)
{
	_flushall();
	printf("Nhap ma so: ");
	gets(sv->ms);
	_flushall();
	printf("Nhap ten:");
	gets(sv->ten);
}
void XuatSV(SINHVIEN sv)
{
	printf("%-5s %-15s\n",sv.ms,sv.ten);
}
void Initialize(LIST *list) {
	list->num = 0;
}
int ListSize(LIST list) {
	return list.num;
}
bool Empty(LIST list)
{
	return list.num == 0 ? true : false;
}
bool Full(LIST list)
{
	return list.num == SIZEMAX ? true : false;
}
```

```
void Traverse(LIST list)
{
	if (Empty(list))
		printf("Danh sach rong\n");
	else
		for (int i = 0; i < list.num; i++)
			XuatSV(list.nodes[i]);
}

void Insert(LIST *list, int pos, SINHVIEN sv)
{
	if (pos < 0 || pos > list->num)
		printf("Vi tri %d khong hop le\n", pos);
	else if (Full(*list))
		printf("Danh sach bi day\n");
	else
	{
		for (int i = list->num - 1; i >= pos; i--)
			list->nodes[i + 1] = list->nodes[i];
		list->nodes[pos] = sv;
		list->num++;
		printf("D/s sinh vien sau khi them SV:\n");
		Traverse(*list);
	}
}
void Remove(LIST *list, int pos)
{
	if (pos < 0 || pos >= list->num)
		printf("Vi tri %d khong hop le\n", pos);
	else if (Empty(*list))
		printf("Danh sach bi rong\n");
	else
	{
		SINHVIEN x = list->nodes[pos];
		for (int i = pos; i<list->num - 1; i++)
			list->nodes[i] = list->nodes[i + 1];
		list->num--;
		printf("Thong tin sinh vien vua bi xoa:\n");
		XuatSV(x);
	}
}
int LinearSearch(LIST list, char *mssv)
{
	for (int i = 0; i < list.num; i++)
		if (_stricmp(list.nodes[i].ms,mssv)==0)
			return i;
	return -1;
}
int BinarySearch(LIST list, char *mssv)
{
	int dau = 0;
	int cuoi = list.num - 1;
	int giua;
	while (dau <= cuoi)
	{
		giua = (dau + cuoi) / 2;
		if (_stricmp(list.nodes[giua].ms,mssv)==0)
			return giua;
		else if (_stricmp(list.nodes[giua].ms,mssv)>0)
			cuoi = giua - 1;
		else
			dau = giua + 1;
	}
	return -1;
}
void Modify(LIST *list)
{
	if (Empty(*list))
		printf("Danh sach bi rong\n");
	else
	{
		char mssv[5];
		_flushall();
		printf("Nhap ma so SV can hieu chinh: ");
		gets(mssv);
		int pos=LinearSearch(*list,mssv);
		if(pos!=-1)
		{
			printf("Nhap ten moi: ");
			gets(list->nodes[pos].ten);
		}
		else
			printf("Khong tim thay SV.");
	}
}

void Sort(LIST *list)
{
	for (int i = 0; i < list->num - 1; i++)
		for (int j = i + 1; j < list->num; j++)
			if (_stricmp(list->nodes[i].ms,list->nodes[j].ms)>0)
			{
				SINHVIEN svt = list->nodes[i];
				list->nodes[i] = list->nodes[j];
				list->nodes[j] = svt;
			}
}
void ClearList(LIST *list)
{
	list->num = 0;
}
void InputList(LIST *list)
{
	do
	{
		printf("Nhap so SV: ");
		scanf("%d", &list->num);
	} while (list->num<1 || list->num>SIZEMAX);
	for (int i = 0; i<list->num; i++)
	{
		printf("Nhap thong tin SV thu %d:\n", i+1);
		NhapSV(&list->nodes[i]);
	}
}

void main()
{
	LIST l;
	int chon;
	Initialize(&l);
	do
	{
		printf("\n-------CHUONG TRINH QUAN LY SINH VIEN SU DUNG D/S
KE------\n");
		printf("1. Them 1 SV vao d/s\n");
		printf("2. Xoa 1 SV khoi d/s\n");
		printf("3. Hieu chinh thong tin SV\n");
		printf("4. Xem d/s SV\n");
		printf("5. Sap xep d/s SV tang dan theo ma so\n");
		printf("6. Tim tuyen tinh SV theo ma so\n");
		printf("7. Tim nhi phan SV theo ma so\n");
		printf("8. Xoa toan bo d/s SV\n");
		printf("0. Thoat CT\n");
		printf("------------------------------------------------------------\n");
		printf("Ban chon: ");
		scanf("%d", &chon);
		switch (chon)
		{
		case 0:
			printf("Dang thoat CT...");
			break;
		case 1:
			SINHVIEN svm;
			printf("Nhap thong tin SV can them:\n");
			NhapSV(&svm);
			int pos;
			printf("Nhap vi tri them SV: ");
			scanf("%d",&pos);
			Insert(&l,pos,svm);
			break;
		case 2:
			if (Empty(l))
				printf("D/s rong");
			else
			{
				printf("Nhap vi tri SV muon xoa: ");
```

```
                        scanf("%d",&pos);
                        Remove(&l,pos);
                        printf("D/s sinh vien sau khi xoa:\n");
                        Traverse(l);
                }
                break;
        case 3:
                if (Empty(l))
                        printf("D/s rong");
                else
                {
                        Modify(&l);
                        printf("D/s sinh vien sau khi hieu chinh:\n");
                        Traverse(l);
                }
                break;
        case 4:
                Traverse(l);
                break;
        case 5:
                if (Empty(l))
                        printf("D/s rong");
                else
                {
                        Sort(&l);
                        printf("D/s SV sau khi sap xep tang dan theo ma so:\n");
                        Traverse(l);
                }
                break;
        case 6:
                if (Empty(l))
                        printf("D/s rong");
                else
                {
                        char mssv[5];
                        _flushall();
                        printf("Nhap ma so SV can tim: ");
                        gets(mssv);
                        int pos=LinearSearch(l,mssv);
                        if(pos!=-1)
                        {
                                printf("Tim thay SV co ma so %s tai vi tri %d:
\n",mssv,pos);
                                XuatSV(l.nodes[pos]);
                        }
                        else
                                printf("Khong tim thay SV co ma so %s trong
d/s\n",mssv);
                }
                break;
        case 7:
                if (Empty(l))
```

```
                        printf("D/s rong");
                else
                {
                        Sort(&l);
                        printf("D/s SV sau khi sap xep tang dan theo ma so:\n");
                        Traverse(l);
                        char mssv[5];
                        _flushall();
                        printf("Nhap ma so SV can tim: ");
                        gets(mssv);
                        int pos=BinarySearch(l,mssv);
                        if(pos!=-1)
                        {
                                printf("Tim thay SV co ma so %s tai vi tri %d:
\n",mssv,pos);
                                XuatSV(l.nodes[pos]);
                        }
                        else
                                printf("Khong tim thay SV co ma so %s trong
d/s\n",mssv);
                }
                break;
        case 8:
                if (Empty(l))
                        printf("D/s rong");
                else
                {
                        ClearList(&l);
                        printf("D/s rong\n");
                }
                break;
        default: printf("Ban chon sai. Moi chon lai");
        }
    } while (chon != 0);
}
```