

DANH SÁCH LIÊN KẾT ĐƠN

(1) Sửa bài tập về nhà buổi 9:

Viết CT sử dụng dslk đơn để quản lý các SV với các chức năng sau:

- a) Xem d/s sinh viên
- b) Thêm sinh viên vào d/s
- c) Xóa sinh viên trong d/s
- d) Cập nhật thông tin sinh viên
- e) Sắp xếp d/s sinh viên theo mã SV
- f) Tìm kiếm sinh viên theo mã SV
- g) Thêm sinh viên vào d/s đã có thứ tự theo mã SV
- h) Xóa toàn bộ d/s sinh viên

```
#define _CRT_SECURE_NO_WARNINGS
#include<stdio.h>
#include<conio.h>
#include<malloc.h>
#include<string.h>

//Khai báo cấu trúc 1 sinh viên
struct Student
{
    char code[5]; //mã SV
    char name[15]; //tên sv
    double mark; //điểm trung bình
    int ship; //học bổng
};

//Khai báo cấu trúc 1 nút SV trong danh sách SV
struct node
{
    Student sv; //chứa thông tin SV
    node *next; //con trỏ chỉ SV kế tiếp trong d/s
};

//Khai báo kiểu con trỏ chỉ đến nút SV
typedef node *NODEPTR;

//Nhập thông tin 1 sinh viên
void InputStudent(Student &sv)
{
    _flushall();
    printf("Nhap masv: ");
    gets(sv.code);
    _flushall();
    printf("Nhap ten sv: ");
    gets(sv.name);
```

```
do
{
    printf("Nhap dtb: ");
    scanf("%lf", &sv.mark);
} while (sv.mark < 0 || sv.mark > 10);
if (sv.mark < 7)
    sv.ship = 0;
else if (sv.mark < 8.5)
    sv.ship = 1000000;
else if (sv.mark < 9)
    sv.ship = 1200000;
else
    sv.ship = 1500000;
}

//Xuất thông tin 1 sinh viên
void OutputSV(Student sv)
{
    printf("%-5s\t%-15s\t%.2lf\t%d\t\n", sv.code, sv.name, sv.mark, sv.ship);
}

//Cấp phát biến động làm 1 nút SV
NODEPTR GetNode()
{
    NODEPTR p;
    p = (NODEPTR)malloc(sizeof(node));
    return p;
}

//Giải phóng biến động làm nút SV đã cấp phát trước đó
void FreeNode(NODEPTR p)
{
    free(p);
}

//Khởi động danh sách SV
void Initialize(NODEPTR *plist)
{
    *plist = NULL;
}

//Kiểm tra d/s có bị rỗng hay không
int Empty(NODEPTR plist)
{
    return (plist == NULL ? 1 : 0);
}
```

```

//Xác định số nút SV có trong d/s
int ListSize(NODEPTR plist)
{
    int count = 0; //biến count (khởi động = 0) để đếm số nút SV trong d/s
    NODEPTR p = plist; //khai báo con trỏ p trỏ đến nút SV đầu d/s
    while (p != NULL) //trong khi d/s có nút SV và chưa đến nút SV cuối
    {
        count++; //tăng giá trị biến count lên 1
        p = p->next; //di chuyển đến nút SV kế
    }
    return count; //trả về số nút SV trong d/s
}

//Xác định con trỏ của nút SV thứ i (i = 0,1,2,...) trong d/s SV
NODEPTR NodePointer(NODEPTR plist, int i)
{
    if (i < 0) //Nếu vị trí i âm
        return NULL;
    else
    {
        NODEPTR p = plist; //khai báo con trỏ p trỏ đến nút SV đầu d/s
        //khai báo biến pos (khởi động = 0) để nhận vị trí nút SV hiện thời
        int pos = 0;
        //trong khi d/s có nút SV và chưa đến nút SV cuối, và chưa đến vị trí thứ i
        while (p != NULL && pos < i)
        {
            p = p->next; //di chuyển p sang nút SV kế
            pos++; //tăng giá trị biến pos lên 1
        }
        return p; //trả về con trỏ nút SV p
    }
}

//Xác định vị trí nút SV p trong d/s SV
int Position(NODEPTR plist, NODEPTR p)
{
    NODEPTR q = plist; //khai báo con trỏ q trỏ đến nút SV đầu d/s
    //khai báo biến pos (khởi động = 0) để nhận vị trí nút SV hiện thời
    int pos = 0;
    //trong khi d/s có nút SV và chưa đến nút SV cuối, và nút SV q không phải nút SV p
    while (q != NULL && q != p)
    {
        q = q->next; //di chuyển q sang nút SV kế
        pos++; //tăng giá trị biến pos lên 1
    }
    if (q != NULL)
        return pos; //vị trí của nút SV p trong d/s
    else
        return -1; //nút SV p không tồn tại
}

```

```

//Xác định nút SV trước nút SV p trong d/s SV
NODEPTR PreNode(NODEPTR plist, NODEPTR p)
{
    if (p == plist)
        return NULL;
    else
    {
        NODEPTR q = plist; //Khai báo con trỏ q trỏ đến nút SV đầu d/s
        //Trong khi chưa hết d/s và nút SV ngay sau q chưa phải là p
        while (q != NULL && q->next != p)
            q = q->next; //di chuyển q sang nút SV kế
        return q; //Trả về q
    }
}

//Thêm sinh viên sv vào đầu d/s SV
void Push(NODEPTR &plist, Student sv)
{
    NODEPTR p = GetNode(); //Cấp phát nút SV p mới
    p->sv = sv; //trường sv của p sẽ lưu thông tin của sinh viên sv
    p->next = plist; //trường next của p sẽ trỏ đến nút SV đầu (plist)
    plist = p; //gán lại vị trí đầu d/s là p
}

//Thêm nút sinh viên q ngay sau nút sinh viên p
void InsAfter(NODEPTR p, Student sv)
{
    if (p == NULL)
        printf("Khong them nut duoc vi d/s rong\n");
    else
    {
        NODEPTR q = GetNode(); //Cấp phát nút SV q mới
        q->sv = sv; //trường sv của q sẽ lưu thông tin của sinh viên sv
        q->next = p->next; //q sẽ trỏ đến nút SV ngay sau nút SV p
        p->next = q; //p sẽ trỏ đến nút SV q
    }
}

//Xoá sinh viên đầu trong d/s
void Pop(NODEPTR &plist)
{
    if (Empty(plist))
        printf("Danh sach bi rong\n");
    else
    {
        NODEPTR p = plist; //Khai báo con trỏ p trỏ đến nút SV đầu d/s
        Student svxoa = p->sv; //svxoa nhận thông tin SV đầu d/s
        plist = p->next; //cho nút SV ngay sau p là đầu d/s
        p->next = NULL; //ngắt kết nối của p với nút ngay sau p
        FreeNode(p); //Giải phóng con trỏ nút p
        printf("\nDa xoa sinh vien dau co thong tin la:\n");
        OutputSV(svxoa); //Xuất thông tin SV đầu đã bị xóa
    }
}

```

```

//Xoá nút SV sau nút SV p
void DelAfter(NODEPTR p)
{
    if (p == NULL || p->next == NULL) //nếu d/s rỗng hoặc p là nút SV cuối
        printf("Khong xoa nut duoc\n");
    else
    {
        NODEPTR q = p->next; //khai báo nút SV q sau nút SV p
        Student svxoa = q->sv; //svxoa nhận thông tin SV nút q
        p->next = q->next; //p sẽ trở đến nút SV ngay sau nút SV q
        q->next = NULL; //ngắt kết nối của q với nút SV ngay sau nó
        FreeNode(q); //Giải phóng con trỏ nút q
        printf("\nDa xoa SV sau co thong tin la:\n");
        OutputSV(svxoa); //Xuất thông tin SV nút q đã bị xóa
    }
}

//Xem danh sách sinh viên
void Traverse(NODEPTR plist)
{
    if (Empty(plist))
        printf("Danh sach bi rong\n");
    else
    {
        NODEPTR p = plist; //Khai báo con trỏ p trở đến nút SV đầu d/s
        while (p != NULL) //trong khi p chưa trở đến nút SV cuối
        {
            OutputSV(p->sv); //Xuất thông tin SV nút p
            p = p->next; //p sẽ trở đến nút SV kế tiếp
        }
    }
}

//Tìm SV theo mã số trong danh sách SV
NODEPTR Search(NODEPTR plist, char *masv)
{
    NODEPTR p = plist; //Khai báo con trỏ p trở đến nút SV đầu d/s
    //Trong khi p chưa trở đến nút SV cuối và mã số của nút SV p khác masv cần tìm
    while (p != NULL && _stricmp(p->sv.code, masv) != 0)
        p = p->next; //di chuyển p sang nút SV kế để tiếp tục tìm
    return p; //trả về nút SV p
}

```

```

//Sắp xếp danh sách SV theo thứ tự tăng dần của mã số SV
void Sort(NODEPTR *plist)
{
    NODEPTR p, q; //Khai báo 2 nút SV p, q
    Student svtemp; //Khai báo cấu trúc svtemp
    //dùng con trỏ p lần lượt duyệt từ nút SV đầu cho đến nút SV kế cuối
    for (p = *plist; p->next != NULL; p = p->next)
    {
        //Dùng con trỏ q lần lượt duyệt từ nút SV sau p cho đến nút SV cuối
        for (q = p->next; q != NULL; q = q->next)
        {
            //Nếu mã SV của nút p đứng sau mã SV của nút q theo thứ tự ABC
            if (_stricmp(p->sv.code, q->sv.code)>0)
            //Hoán vị 2 nút SV p và q
            {
                svtemp = p->sv;
                p->sv = q->sv;
                q->sv = svtemp;
            }
        }
    }
}

//Thêm sinh viên svm vào danh sách SV đã có thứ tự theo mã SV
void Place(NODEPTR *plist, Student svm)
{
    NODEPTR p, q = NULL; //Khai báo 2 nút SV p (trỏ đến nút SV đầu d/s) và q
    //Duyệt từ đầu d/s để tìm SV ngay trước SV cần thêm
    //Trong khi p chưa trỏ đến nút SV cuối và mã số của nút SV p khác masv cần tìm
    for (p = *plist; p != NULL && _stricmp(p->sv.code, svm.code)<0; p = p->next)
        q = p; //Nút SV q trỏ nút p, và nút p trỏ đến nút kế
    if (q == NULL) //Nếu chưa có nút SV nào trong d/s
        Push(*plist, svm); //gọi hàm thêm sinh viên svm vào đầu d/s
    else
        InsAfter(q, svm); //gọi hàm thêm sinh viên svm vào sau sinh viên q
}

//Xoá danh sách sinh viên
void ClearList(NODEPTR *plist)
{
    NODEPTR p = *plist, q = NULL; //Khai báo 2 nút SV p (trỏ đến nút SV đầu d/s) và q
    while (p != NULL) //trong khi chưa đến cuối d/s
    {
        q = p; //xác định nút hiện hành
        p = p->next; //di chuyển đến nút kế tiếp
        FreeNode(q); //giải phóng nút hiện hành
    }
    Initialize(plist); //khởi động lại ds rỗng
}

```

```

void main()
{
    NODEPTR plist,p;
    Student sv;
    int chon;
    char tl;
    Initialize(&plist); //Gọi hàm khởi động d/s SV
    do
    {
        printf("\n--CHUONG TRINH QUAN LY SINH VIEN SU DUNG DSLK DON--\n");
        printf("1. Xem danh sach SV\n");
        printf("2. Them SV vao danh sach\n");
        printf("3. Xoa SV trong danh sach\n");
        printf("4. Cap nhat thong tin SV\n");
        printf("5. Sap xep danh sach SV tang dan theo masv\n");
        printf("6. Tim SV theo masv\n");
        printf("7. Them SV vao d/s da duoc sap thu tu theo masv\n");
        printf("8. Xoa 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:
                if (Empty(plist))
                    printf("D/s SV rong\n");
                else
                {
                    printf("Danh sach SV hien tai la:\n");
                    Traverse(plist); //xuất d/s SV
                }
                break;
            case 2:
                printf("Nhap thong tin SV:\n");
                InputStudent(sv); //gọi hàm nhập thông tin 1 SV
                int vtp; //nhận giá trị nhập vị trí muốn thêm SV vào d/s
                printf("Nhap vi tri muon them SV: ");
                scanf("%d", &vtp);
                if (Empty(plist) || vtp == 0) //nếu d/s rỗng hoặc vị trí muốn thêm vtp là 0
                    Push(plist, sv); //gọi hàm thêm sinh viên sv vào đầu dslk
                else
                {
                    //Xác định con trỏ nút SV p trước vị trí muốn thêm trong d/s
                    p = NodePointer(plist, vtp - 1);
                    if (p != NULL) //nếu tồn tại nút SV p trước vị trí muốn thêm
                        InsAfter(p, sv); //Gọi hàm thêm sinh viên sv sau nút SV p
                    else
                    {
                        printf("Vi tri khong hop le.");
                        break;
                    }
                }
                printf("Danh sach SV hien tai la:\n");
                Traverse(plist); //xuất d/s SV
                break;
        }
    }
}

```

```

case 3:
    if (Empty(plist))
        printf("D/s rong\n");
    else
    {
        int vtx;//nhận giá trị nhập vị trí muốn xóa SV trong d/s
        printf("Nhap vi tri SV muon xoa: ");
        scanf("%d",&vtx);
        if (vtx == 0)//nếu vị trí muốn xóa vtx là 0
            Pop(plist);//gọi hàm xóa sinh viên đầu d/s
        else
        {
            //Xác định con trỏ nút SV p trước vị trí muốn xóa trong d/s
            p = NodePointer(plist, vtx-1);
            DelAfter(p);//gọi hàm xóa nút SV sau nút SV p
        }
        printf("Danh sach SV hien tai la:\n");
        Traverse(plist); //xuất d/s SV
    }
    break;
case 4:
    if (Empty(plist))
        printf("D/s rong\n");
    else
    {
        int vts;//nhận giá trị nhập vị trí muốn cập nhật thông tin SV trong d/s
        printf("Nhap vi tri SV muon cap nhat thong tin: ");
        scanf("%d", &vts);
        //Xác định con trỏ nút SV p tại vị trí muốn cập nhật vts
        p = NodePointer(plist, vts);
        if (p != NULL)//nếu con trỏ nút SV p tồn tại
        {
            printf("Nhap thong tin moi:\n");
            InputStudent(sv);//gọi hàm nhập mới thông tin sinh viên sv
            p->sv = sv;//cập nhật trường sv của nút SV p theo thông tin mới
            printf("Danh sach SV hien tai la:\n");
            Traverse(plist); //xuất d/s SV
        }
        else
            printf("Vi tri khong hop le.");
    }
    break;
case 5:
    if (Empty(plist))
        printf("D/s rong\n");
    else
    {
        Sort(&plist);//gọi hàm sắp xếp d/s SV theo thứ tự mã số SV tăng dần
        printf("Danh sach SV hien tai la:\n");
        Traverse(plist); //xuất d/s SV
    }
    break;

```



```

case 6:
    if (Empty(plist))
        printf("D/s rong\n");
    else
    {
        _flushall();
        printf("Nhap ma SV can tim: ");
        gets(sv.code);
        p = Search(plist,sv.code);//Goi ham tim nut SV p co ma so can tim
        if (p != NULL)//neu nut SV p ton tai
        {
            int posp = Position(plist, p);//bien posp nhan vi tri nut p
            printf("Tim thay tai vi tri thu %d\n", posp);
        }
        else
            printf("Khong tim thay.");
    }
    break;
case 7:
    printf("Nhap thong tin SV muon them:\n");
    InputStudent(sv);
    if (Empty(plist))//neu d/s SV rong
        Push(plist, sv);//goi ham them sinh vien sv vào đầu d/s SV
    else
        Place(&plist,sv);//hoi ham them SV vào d/s đã có thứ tự theo mã SV
    printf("Danh sach SV hien tai la:\n");
    Traverse(plist); //xuất d/s SV
    break;
case 8:
    if (Empty(plist))
        printf("D/s rong\n");
    else
    {
        printf("Ban muon xoa d/s SV? (C/K) ");
        tl = _getch();
        if (tl == 'C' || tl == 'c')
        {
            ClearList(&plist);//goi ham xoa
            printf("\nDa xoa d/s SV\n");
        }
    }
    break;
default: printf("Ban chon sai. Moi chon lai.");
}
} while (chon != 0);
_getch();
}

```