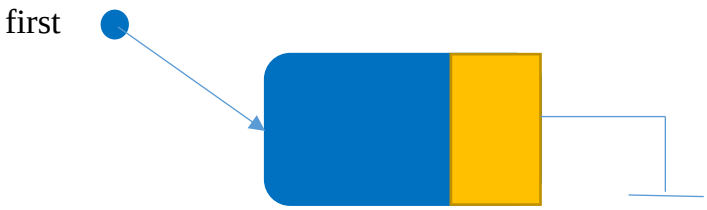
BÀI THỰC HÀNH SỐ 2

1. MỤC TIÊU:

- Làm quen với cấu trúc dữ liệu: danh sách liên kết đơn
- Thực hiện các thao tác cơ bản trên danh sách liên kết đơn:
  - Tạo danh sách
  - Thêm mới phần tử vào vào danh sách
  - Duyệt/tìm kiếm các phần tử trên danh sách
  - Xóa phần tử trên danh sách

2. LÝ THUYẾT CẦN GHI NHỚ

- Danh sách liên kết gồm 1 hoặc nhiều phần tử, mỗi phần tử là một node có cấu trúc gồm 2 thành phần (info và link), trong đó info là thành phần chứa thông tin, link chứa địa chỉ của phần tử tiếp theo (hoặc NULL nếu như không có phần tử tiếp theo)
- Mỗi danh sách được quản lý bởi con trỏ first. Con trỏ này chứa địa chỉ của phần tử đầu tiên trong danh sách



3. BÀI TẬP THỰC HÀNH CƠ BẢN

SV tạo project "Win32 Console Application" trên VS 2015, với tên là "**Lab2**".

Tạo file source .cpp có tên: **hovaten_mssv_lab2.cpp**

*** Cách tạo project và cấu trúc chương trình C++: Xem lại file hướng dẫn thực hành lab 1

Bài 1:

Quản lý một danh sách có số phần tử khá lớn, biến động. Mỗi phần tử có kiểu int. (Dùng cấu trúc danh sách liên kết đơn)

a. Khai báo cấu trúc danh sách

Để khao báo cấu trúc danh sách cần khai báo cấu trúc của 1 phần tử trong danh sách và khai báo con trỏ first.

```
#include <iostream>
using namespace std;
// khai bao struct Node gom 2 thanh phan: info va link
struct Node {
    int info;
    Node* link;
};
Node* first; // contro first
```

b. Viết thủ tục khởi tạo danh sách rỗng

Danh sách rỗng là danh sách ban đầu chưa có phần tử nào. Do đó con trỏ first được gán giá trị là NULL

```
void init() {
    first = NULL;
}
```

c. Viết thủ tục thêm một phần tử vào đầu danh sách, với giá trị thêm vào vùng info là x có kiểu int

Các bước thực hiện code:

- Tạo 1 node p mới (p gồm có info và link)
- Gán x vào vùng info của node
- Gán thành phần link của p có giá trị NULL
- Trỏ con trỏ first vào p

Cách 1:

```
void Insert_first(int x)
{
    Node* p;
    p = new Node;
    p->info = x;
    p->link = first; // do first ban dau co gia tri null
    first = p;
}
```

Cách 2: (cách này phải viết 2 hàm)

- Viết thủ tục tạo node p
- Trong thủ tục Insert_first thì trỏ con trỏ first về node vừa mới tạo

```
/// Thu tuc tao 1 node gom 2 thanh phan, gan x vao info va NULL cho link
/// </summary>
Node* createNode(int x) {
    Node* p;
    p = new Node;
    p->info = x;
    p->link = NULL;
    return p;
}
// tro con tro first ve node vua moi tao
void Insert_first_2(int x) {
    Node* p = createNode(x);
    p->link = first;
    first = p;
}
```

d. Viết thủ tục xuất các phần tử trong danh sách

- Danh sách liên kết đơn không giới hạn số lượng phần tử, do đó không thể dùng vòng for để duyệt và xuất các phần tử trong danh sách.
- Một phần tử trong danh sách nếu là phần tử cuối cùng thì thành phần link của nó sẽ có giá trị NULL. Do đó để duyệt và xuất các phần tử trong danh sách cần dùng vòng lặp while

```
void Process_List() {
    Node* p;
    p = first; // gan p tro den phan tu dau tien
    while (p != NULL) {
        cout << p->info << "\t";
        p = p->link; // p tro den phan tu tiep theo
    }
    cout << endl;

}
```

e. Viết thủ tục tìm một phần tử trong danh sách.

- Để tìm phần tử x trong một danh sách, cần duyệt danh sách từ phần tử đầu tiên cho đến khi tìm được ➔ dùng vòng lặp while để duyệt

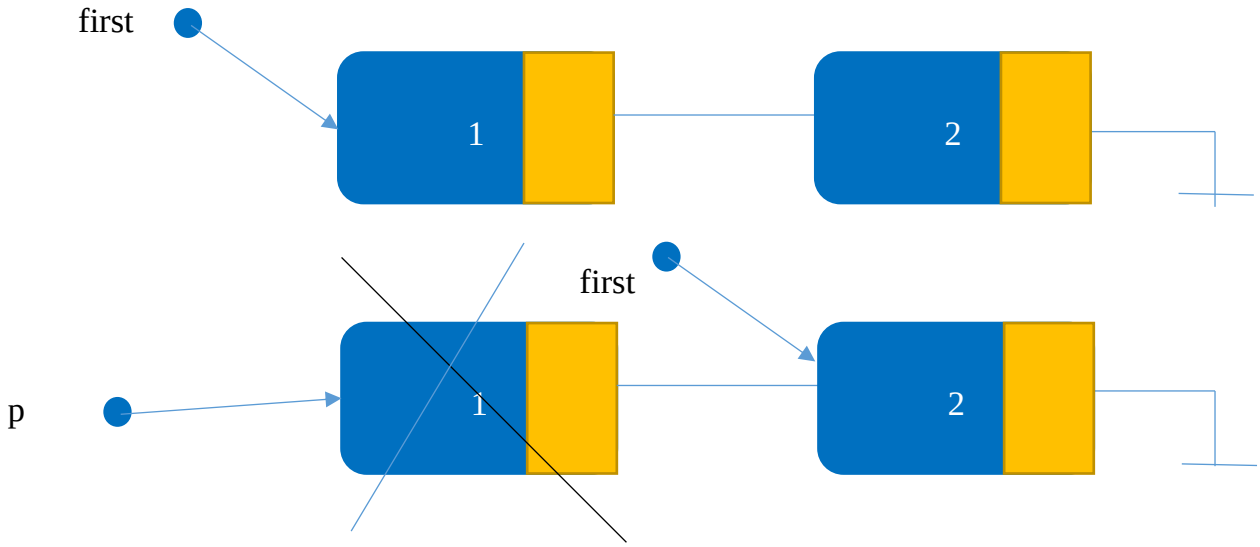
```
Node* Search(int x)
{
    Node* p = first;
    while ((p != NULL) && (p->info != x))
        p = p->link;
    return p; // neu khong tim thấy thì p có giá trị là NULL
}
```

f. Viết thủ tục xóa một phần tử dầu danh sách.

- Để xóa phần tử đầu tiên trong danh sách:
  - Cô lập phần tử đầu tiên bằng cách tạo con trỏ p trỏ đến phần tử đầu tiên trong danh sách, sau đó con trỏ first sẽ trỏ đến phần tử thứ 2
  - Xóa phần tử đầu tiên bằng cách xóa con trỏ p

*** Lưu ý: Con trỏ first luôn trỏ đến phần tử đầu tiên của danh sách, do đó không được xóa con trỏ first.

-




```
int Delete_first()
{
    if (first != NULL) // danh sách khác rỗng
    {
        Node* p = first;
        first = first->link;
        delete p;
        return 1;
    }
    return 0;
}
```

g. Viết thủ tục thêm một phần tử vào cuối danh sách

- Để thêm phần tử vào cuối danh sách:
  - Tìm phần tử n cuối của danh sách (dùng vòng while duyệt)
  - n->link = p (p là phần tử cần thêm)
- SV tự viết code

```
void insert_last(int x) {
    Node* newNode = createNode(x);
    if (first == NULL) // SV tu viet code
    else {
        Node* p = new Node;
        p = first;
        while (p->link != NULL)
            // SV tu viet code
        p->link = newNode;
    }
}
```

h. Viết thủ tục xóa một phần tử cuối danh sách

- Để xóa phần tử cuối danh sách:
  - Tìm phần tử kế cuối và phân tử cuối (dùng vòng while để duyệt)
  - Cho thành phần link của phần tử kế cuối trỏ về NULL
  - Xóa phần tử cuối
- SV tự viết code

i. Viết thủ tục tìm một phần tử trong danh sách. Nếu tìm thấy, hãy xóa phần tử này

- Tìm phần tử trong ds (câu e)
- Xác định phần tử này:
  - Là phần tử đầu ➔ câu f
  - Phần tử cuối ➔ câu h
  - Là phần tử giữa:
    - Các định phần tử trước nó
    - Thay đổi liên kết
    - Xóa
- SV tự viết code

j. Từ danh sách trên hãy chuyển thành danh sách có thứ tự (*)

- SV tự viết code

k. Viết thủ tục xóa tất cả các phần tử trong danh sách liên kết đơn

- SV tự viết code

Bài 2:

Quản lý một danh sách có thứ tự, số phần tử khá lớn, biến động. Mỗi phần tử có kiểu int. (Dùng cấu trúc danh sách liên kết đơn)

a. Khai báo cấu trúc danh sách (tương tự cầu 1a)
b. Viết thủ tục khởi tạo danh sách rỗng. (tương tự câu 1b)
c. Viết thủ tục thêm một phần tử vào danh sách (thêm một phần tử vào vị trí phù hợp trong danh sách đã có thứ tự). Lưu ý: Không xếp thứ tự danh sách.
d. Viết thủ tục xuất các phần tử trong danh sách. (tương tự câu 1 d)

e. Viết thủ tục tìm một phần tử trong danh sách (lưu ý: danh sách đã có thứ tự) Tương tự như tìm trong ds liên kết đơn chưa sắp xếp thứ tự, nhưng điều kiện tìm là giá trị của current node < giá trị tìm

f. Viết thủ tục tìm một phần tử trong danh sách. Nếu tìm thấy, xóa phần tử này (Lưu ý: danh sách đã có thứ tự) (tương tự câu 1i)

4. BÀI TẬP NÂNG CAO

Bài 1:

- Viết thủ tục chèn node có giá trị x vào phía sau node có giá trị lớn nhất

Bài 2:

- Viết thủ tục cho biết các danh sách có toàn số chẵn/lẽ hay không
    - Mở rộng: cho biết danh sách có chứa toàn số chính phương/nguyên tố không

5. BÀI TẬP VỀ NHÀ

Bài 1:

- Viết thủ tục cho biết các phần tử chẵn/lẻ có xuất hiện xen kẻ trong danh sách liên kết đơn hay không