代码：

#include <iostream>

using namespace std;

class Node {

private:

int Number;

Node\* Next;

Node\* Last;

public:

Node(const int& n);

const int& GetNumber() const {

return Number;

}

const Node\* GetNext()const {

return Next;

}

const Node\* GetLast()const {

return Last;

}

friend class LinkedList;

};

Node::Node(const int& n) {

this->Number = n;

this->Next = nullptr;

this->Last = nullptr;

}

class LinkedList {

private:

Node\* Head;

Node\* Tail;

public:

LinkedList() {

Head = nullptr;

Tail = nullptr;

}

void Clear();//清除堆空间

~LinkedList() { Clear(); }//析构函数

const Node\* GetHead()const { return Head; }

Node\* Add(const int& Data);//插入在尾部

const int GetCount();//计总数

const int GetThisCount(Node\* This); //计当前位置

int CheckMaxPlace();//查找最大数位置

};

void LinkedList::Clear() {

while (Head) {

Node\* newHead = Head->Next;

delete Head;

Head = newHead;

}

}

Node\* LinkedList::Add(const int& Data) {

Node\* newNode = new Node(Data);

if (!Tail)

Head = Tail = newNode;

else

{

Tail->Next = newNode;

Tail = newNode;

}

return newNode;

}

const int LinkedList::GetCount() {

int i = 0;

for (Node\* Count = Head; Count; Count = Count->Next)i++;

return i;

}

const int LinkedList::GetThisCount(Node\* This) {

int i = 1;

for (Node\* Count = Head; (Count)&&(Count != This); Count = Count->Next)i++;

return i;

}

int LinkedList::CheckMaxPlace() {

int n = Head->Number,i=1;

Node\* This= nullptr ;

for (Node\* Count = Head->Next; Count; Count = Count->Next) {

if (n < Count->Number) {

n = Count->Number;

This = Count;

}

}

i = GetThisCount(This);

return i;

}

ostream& operator << (ostream& o, const LinkedList& list)

{

for (const Node\* p = list.GetHead();

p;

p = p->GetNext())

{

o << "[" << p->GetNumber() << "]";

if (p->GetNext())

o << " -> ";

}

return o;

}

int main()

{

LinkedList Mylist;

Mylist.Add(2);

Mylist.Add(1);

Mylist.Add(7);

Mylist.Add(16);

Mylist.Add(22);

Mylist.Add(0);

Mylist.Add(33);

Mylist.Add(12);

Mylist.Add(18);

cout << "Mylist is" << endl;

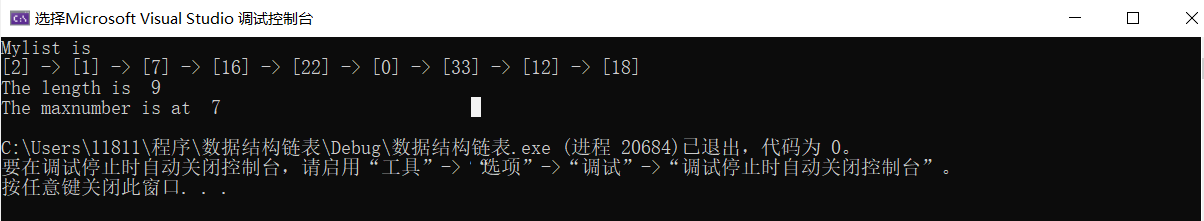
cout << Mylist<< endl;

cout << "The length is " << Mylist.GetCount()<< endl;

cout << "The maxnumber is at " << Mylist.CheckMaxPlace() << endl;

}

运行结果：



时间复杂度：

添加n个数字：3n；计算总长度：n；输出链表：2n-1；

查找最大数位置的时间复杂度，假设每个位置出现最大数的可能性相同，并且假设找到最大数前，不同置换次数的可能性相同：

只在第a个位置置换一次最大数所需语句数：2+a

在第a个位置置换一次前已经置换一次经过的语句数：2+2+a

在第a个位置置换一次前已经置换k次的语句数：2+2k+a

到第a位停止(即a最大)，所有等可能语句总数：

最大值等可能出现在n个位置：

总语句：3n+n+2n-1+n-2=6n-3，即O（n）；