A diagram of a graph

Description automatically generated with medium confidence

#include <iostream>

using namespace std;

class CNode

{

public:

int info;

CNode\* pNext;

};

class CList

{

public:

CNode\* pHead;

CNode\* pTail;

CList()

{

pHead = NULL;

pTail = NULL;

}

void Attach(CNode\* pnn)

{

if (pHead == NULL)

{

pHead = pnn;

pTail = pnn;

}

else

{

pTail->pNext = pnn;

pTail = pnn;

}

}

~CList()

{

CNode\* pTrav = pHead;

while (pHead != NULL)

{

pHead = pTrav->pNext;

pTrav->pNext = NULL;

delete pTrav;

pTrav = pHead;

}

}

};

void main()

{

CList L;

CNode\* pnn;

int N,V,pos=0,check=0,ctb=0,tot=0;

cout << "Enter N \n";

cin >> N;

for (int i = 0; i < N; i++)

{

pnn = new CNode;

cout << "enter info\n";

cin >> pnn->info;

pnn->pNext = NULL;

L.Attach(pnn);

}

cout << "Enter value \n";

cin >> V;

CNode\* pTrav = L.pHead;

for (int i = 0; i < N; i++)

{

if (pTrav->info == V)

{

pos = i;

check = 1;

}

if ((check==1 && i == pos + 1) ||(check==1 && i == pos + 2) ||(check==1 && i == pos + 3))

{

tot+= pTrav->info;

}

pTrav = pTrav->pNext;

}

pTrav = L.pHead;

for (int i = 0; i < pos; i++)

{

if (i == pos - 1 || i == pos - 2 || i == pos - 3)

{

tot+= pTrav->info;

}

pTrav = pTrav->pNext;

}

cout <<"total:"<< tot;

}

A diagram of a graph

Description automatically generated

#include <iostream>

using namespace std;

class CNode

{

public:

int info;

CNode\* pNext;

};

class CList

{

public:

CNode\* pHead;

CNode\* pTail;

CList()

{

pHead = NULL;

pTail = NULL;

}

void Attach(CNode\* pnn)

{

if (pHead == NULL)

{

pHead = pnn;

pTail = pnn;

}

else

{

pTail->pNext = pnn;

pTail = pnn;

}

}

~CList()

{

CNode\* pTrav = pHead;

while (pHead != NULL)

{

pHead = pTrav->pNext;

pTrav->pNext = NULL;

delete pTrav;

pTrav = pHead;

}

}

};

void main()

{

CList L1;

CList L2;

CNode\* pn1;

CNode\* pn2;

int N;

cout << "Enter N \n";

cin >> N;

for (int i = 0; i < N; i++)

{

pn1 = new CNode;

cout << "enter info\n";

cin >> pn1->info;

pn1->pNext = NULL;

L1.Attach(pn1);

}

for (int i = 0; i < N; i++)

{

pn2 = new CNode;

cout << "enter info\n";

cin >> pn2->info;

pn2->pNext = NULL;

L2.Attach(pn2);

}

CNode\* pTrav1 = L1.pHead;

CNode\* pTrav2 = L2.pHead;

for (int i = 0; i < N; i++)

{

if (pTrav1->info == pTrav2->info)

{

cout << "[ " << i << " , " << pTrav1->info << " ]\n";

}

pTrav1 = pTrav1->pNext;

pTrav2 = pTrav2->pNext;

}

}