**Section (I): Tracing Problems (Total: 3 marks)**

*In the following tracing question use the following definition for the nodes in the list:*

**class CListNode**

pNext

info

pBack

**{**

**public:**

**int info;**

**CListNode \*pNext;**

**CListNode \*pBack;**

**};**

**A**

**Write one statement to do the followings:-**

* Display the info of node (1).

Cout<<A->pNext->pNext->info;

* Display the info of node (2).

Cout<<A->pBack->info;

* Make the “pBack” of Node(3) points to the Node (2).

A->pNext->pBack=A->pBack;

**Section (II): Algorithms (Total: 7 marks)**

**Algorithm 1: (7 marks)**

*In the following Algorithm question use the following definition for the nodes in the list:*

**class CNode**

info

**{**

**public:**

pNext

**int info;**

**CNode \*pNext;**

**};**

**class CStack**

**{**

H1

**public:**

H3

**CNode \*H1, \*H2, \*H3, \*H4;**

H4

H2

**…………………………..**

**…………………………..**

**};**

Write a function with **O(1)** which called

*CNode\** ***Pop\_And\_Rotate*** *( )*

* *The function will pop the node from (H1)*
* *Rotate the first nodes (H3) 🡪 (H4) 🡪 (H2) 🡪 (H1)*
* *Return the popped node from (H1)*

H1

H3

H4

H2

The returned node

H4

H3

H2

H1

CNode\* Pop\_And\_Rotate()

{

CNode\* p1, \* p2, \* p3, \*p4;

p1 = H1, p2 = H2, p3 = H3, p4 = H4;

H3 = p3->pNext;

p3->pNext = NULL;

H4 = p4->pNext;

p4->pNext = NULL;

H2 = p2->pNext;

p2->pNext = NULL;

H1 = p1->pNext;

p1->pNext = NULL;

p3->pNext = H4;

H4 = p3;

p4->pNext = H2;

H2 = p4;

p2->pNext = H1;

H1 = p2;

return p1;

}

**Section (III): Problem Solving (Total: 10 marks)**

**Problem 1: (10 marks)**

pDown

info

**class CNode**

pNext

**{**

**public:**

**int info;**

**CNode \* pNext;**

**CNode \* pDown;**

**};**

Write a main function to do:

* + Read 5 Lists from the user.

(Assume all lists with the same length)

Head

Head

Head

L

Head

Head

* Find the zero-intervals and connect the last node of each interval to the first node of the next interval.
  + Use the pDown for the connection process.

.

Head

L

Head

Head

Head

Head

void main()

{

CList L[5];

CNode\* pnn, \* pTrav, \*pF, \*pL;

int N, max = -9999, min = 9999, ct = 0, posmin = 0, posmax = 0;

cout << "enter n \n";

cin >> N;

for (int i = 0; i < 5; i++)

{

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

{

pnn = new CNode;

cout << "enter pnn info \n";

cin >> pnn->info;

pnn->pNext = NULL;

pnn->pDown = NULL;

L[i].Attach(pnn);

}

}

for (int i = 0; i < 5; i++)

{

pTrav = L[i].pHead;

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

{

if (pTrav->info == 0 && (pTrav->pNext->info != 0 || pTrav->pNext == NULL))

{

pL = pTrav;

}

else

{

if (pTrav->info != 0 && pTrav->pNext->info == 0 && i>0)

{

pF = pTrav->pNext;

pL->pDown = pF;

}

}

pTrav = pTrav->pNext;

}

}

}