**20145165 정균모 자료구조 과제**

**public** **class** Node {

**int** data;

Node prevNode;

Node nextNode;

**public** Node(**int** data) {

**this**.data = data;

}

}

**public** **class** CircularDoubleLinkedList {

**private** Node head;

**public** **void** add(Node newNode) {

**if**(head == **null**) {

head = newNode;

head.prevNode = head;

head.nextNode = head;

}**else** {

Node tail = head.prevNode;

tail.nextNode = newNode;

head.prevNode = newNode;

newNode.prevNode = tail;

newNode.nextNode = head;

}

}

**public** **void** add(Node newNode, **int** location) {

**if**(location == 0) {

newNode.nextNode = head;

newNode.prevNode = head.prevNode;

head.prevNode = newNode;

newNode.prevNode.nextNode = newNode;

head = newNode;

}**else** {

Node before = head;

**while**((--location) > 0) {

before = before.nextNode;

}

newNode.prevNode = before;

newNode.nextNode = before.nextNode;

before.nextNode = newNode;

newNode.nextNode.prevNode = newNode;

}

}

**public** Node get(**int** location) {

Node current = head;

**while**((--location) >= 0) {

current = current.nextNode;

}

**return** current;

}

**public** **void** remove(**int** location) {

**if**(location == 0) {

head.prevNode.nextNode = head.nextNode;

head.nextNode.prevNode = head.prevNode;

head = head.nextNode;

}**else** {

Node before = head;

**while**((--location) > 0) {

before = before.nextNode;

}

before.nextNode = before.nextNode.nextNode;

before.nextNode.prevNode = before;

}

}

**public** **int** size() {

Node current = head;

**int** size = 0;

**while**(current != **null**) {

current = current.nextNode;

size++;

**if**(current == head) {

**break**;

}

}

**return** size;

}

}

**public** **class** Test\_CircularLinkedList {

**public** **static** **void** main(String[] args) {

CircularDoubleLinkedList list = **new** CircularDoubleLinkedList();

System.***out***.println("20145165 정균모");

System.***out***.println("노드추가");

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

list.add(**new** Node(i));

}

System.***out***.print("[ ");

**for**(**int** i = 0; i < list.size(); i++) {

System.***out***.print(list.get(i).data + " ");

}

System.***out***.println("]");

System.***out***.println("노드 인덱스 2를 삭제");

list.remove(2);

System.***out***.print("[ ");

**for**(**int** i = 0; i < list.size(); i++) {

System.***out***.print(list.get(i).data + " ");

}

System.***out***.println("]");

System.***out***.println("노드 인덱스 2에 10을 추가");

list.add(**new** Node(10), 2);

System.***out***.print("[ ");

**for**(**int** i = 0; i < list.size(); i++) {

System.***out***.print(list.get(i).data + " ");

}

System.***out***.println("]");

System.***out***.println("원형리스트확인을 위해 2바퀴 출력");

System.***out***.print("[ ");

**for**(**int** i = 0; i < list.size() \* 2; i++) {

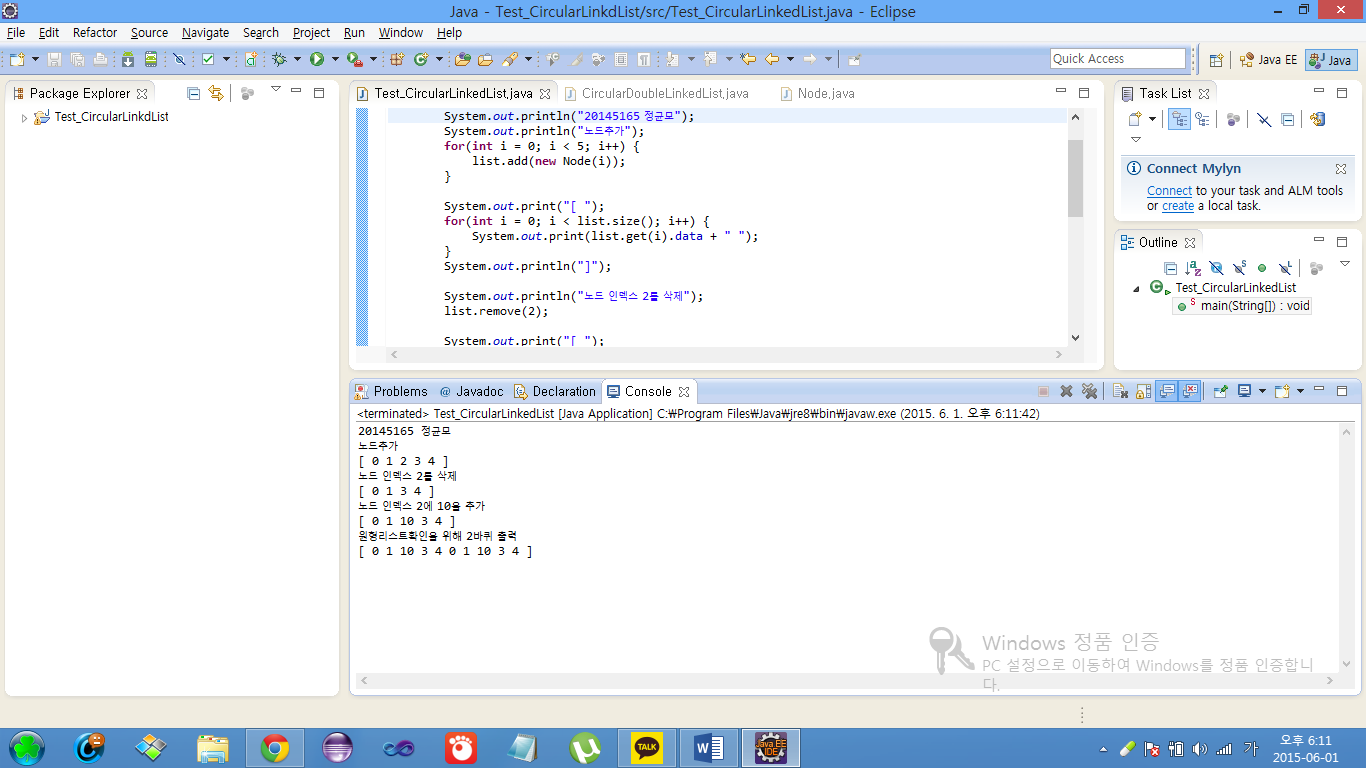
System.***out***.print(list.get(i).data + " ");

}

System.***out***.println("]");

}

}



**public** **class** PolyNode {

**private** **int** coef;

**private** **int** exp;

PolyNode link;

**public** PolyNode(**int** c, **int** e) {

**this**.coef = c;

**this**.exp = e;

}

**public** **int** getCoef() {

**return** coef;

}

**public** **int** getExp() {

**return** exp;

}

}

**public** **class** Polynomial22 {

**private** PolyNode firstNode;

**private** PolyNode LastNode;

**public** **boolean** ispZero() {

**if**(firstNode != **null**) {

**return** **true**;

}**else** {

**return** **false**;

}

}

**public** **void** addTerm(**int** c, **int** e) {

PolyNode node = **new** PolyNode(c, e);

**if**(firstNode == **null**) {

firstNode = node;

LastNode = node;

}**else** {

LastNode.link = node;

LastNode = node;

}

}

**public** **void** print() {

PolyNode p = firstNode;

**while**(p != **null**) {

**if**(p.link == **null**) {

System.***out***.println(p.getCoef() + "x^" + p.getExp());

} **else** {

System.***out***.print(p.getCoef() + "x^" + p.getExp() + "+");

}

p = p.link;

}

}

**public** Polynomial22 sMult(**int** c, **int** e) {

PolyNode p = **this**.firstNode;

Polynomial22 res = **new** Polynomial22();

**while**(p != **null**) {

**int** mult = p.getCoef() \* c;

**if**(mult != 0) {

res.addTerm(mult, p.getExp() + e);

}

p = p.link;

}

**return** res;

}

**public** Polynomial22 PolyAdd(Polynomial22 poly) {

PolyNode p = **this**.firstNode;

PolyNode q = poly.firstNode;

Polynomial22 res = **new** Polynomial22();

**while**(p != **null** && q != **null**) {

**if**(p.getExp() == q.getExp()) {

**int** sum = p.getCoef() + q.getCoef();

**if**(sum != 0) {

res.addTerm(sum, p.getExp());

}

p = p.link;

q = q.link;

} **else** **if**(p.getExp() < q.getExp()) {

res.addTerm(q.getCoef(), q.getExp());

q = q.link;

} **else** {

res.addTerm(p.getCoef(), p.getExp());

p = p.link;

}

}

**while**(p != **null**) {

res.addTerm(p.getCoef(), p.getExp());

p = p.link;

}

**while**(q != **null**) {

res.addTerm(q.getCoef(), q.getExp());

q = q.link;

}

LastNode.link = **null**;

**return** res;

}

}

**public** **class** Ex2 {

**public** **static** **void** main(String[] args) {

System.***out***.println("20145169 정용석");

System.***out***.println("Create Nodes for Polynomial A");

Polynomial22 A = **new** Polynomial22();

A.addTerm(6, 5);

A.addTerm(3, 3);

A.addTerm(9, 2);

A.addTerm(2, 1);

System.***out***.print("A = ");

A.print();

System.***out***.println("Create Nodes for Polynomial B");

Polynomial22 B = **new** Polynomial22();

B.addTerm(2, 7);

B.addTerm(5, 5);

B.addTerm(10, 2);

B.addTerm(3, 1);

System.***out***.print("B = ");

B.print();

System.***out***.println("Create Nodes for Polynomial A+B");

Polynomial22 C = **new** Polynomial22();

C = A.PolyAdd(B);

System.***out***.print("A+B = ");

C.print();

System.***out***.println("Create B\* 3^2 sMult");

Polynomial22 D = **new** Polynomial22();

D = B.sMult(3, 2);

System.***out***.print("D = ");

D.print();

}

}

