



YONSEI UNIVERSITY

Homework #2

Data Structure



- Data : n 개의 element로 구성된 순서 있는 모임
- Operations :
 - insertFirst(list, item) : 맨 앞에 요소(item) 삽입
 - insertLast(list, item) : 맨 끝에 요소(item) 삽입
 - Insert(list, pos, item) : pos 위치에 요소(item) 삽입
 - delete(list, pos) : pos 위치의 요소(item) 삭제
 - clear(list) : 모든 요소 삭제
 - replace(list, pos, item) : pos 위치의 요소를 item으로 교체
 - isList(list, item) : item이 리스트에 있는지 검사
 - getItem(list, pos) : pos 위치의 요소를 반환
 - getLength(list) : 리스트의 길이(항목의 개수)를 구함
 - isEmpty(list) : 리스트가 비었는지 검사
 - isFull(list) : 리스트가 꽉 찼는지 검사
 - display(list) : 리스트의 모든 요소를 표시

HW#2.1. Linear List 구현 (HW2_LinearList.c)

■ 아래와 같이 실행되도록 main()함수 구성

- 리스트 초기화
- 0에 10 삽입
- 0에 20 삽입
- 맨 앞에 5 삽입
- 맨 끝에 30 삽입
- -1에 3 삽입

• 현재 리스트 모든 항목 보여주기

• 2의 항목 삭제

• 현재 리스트 모든 항목 보여주기

• 1의 항목을 50으로 교체

• 현재 리스트 모든 항목 보여주기

■ List ADT의 모든 연산 구현

```
E:\Lecture\2020-1\2020-1 데이터구조론\Src\linear...
ArrayList[0]에 10 삽입
ArrayList[0]에 20 삽입
ArrayList[0]에 5 삽입
ArrayList[3]에 30 삽입
Index Error
===Print Lists===
ArrayList[0]: 5
ArrayList[1]: 20
ArrayList[2]: 10
ArrayList[3]: 30
ArrayList[2]의 10 삭제
===Print Lists===
ArrayList[0]: 5
ArrayList[1]: 20
ArrayList[2]: 30
===Print Lists===
ArrayList[0]: 5
ArrayList[1]: 50
ArrayList[2]: 30
-----
Process exited after 0.381 seconds with return value 0
계속하려면 아무 키나 누르십시오 . . .
```

■ 다항식의 덧셈 연산(addPoly(A,B)) 알고리즘

addPoly(A, B)

```
C ← zeroP();
while (not isZero(A) and not isZero(B)) do
  if maxExp(A) < maxExp(B) then
    C ← addTerm(C, coef(B, maxExp(B)), maxExp(B));
    B ← delTerm(B, maxExp(B));
  else if maxExp(A) = maxExp(B) then
    sum ← coef(A, maxExp(A)) + coef(B, maxExp(B));
    if sum ≠ 0 then C ← addTerm(C, sum, maxExp(A));
    A ← delTerm(A, maxExp(A));
    B ← delTerm(B, maxExp(B));
  else then
    C ← addTerm(C, coef(A, maxExp(A)), maxExp(A));
    A ← delTerm(A, maxExp(A));
  endif
endwhile

if (not isZero(A)) then A의 나머지 항들을 C에 복사
else if (not isZero(B)) then B의 나머지 항들을 C에 복사
endif

return C
```

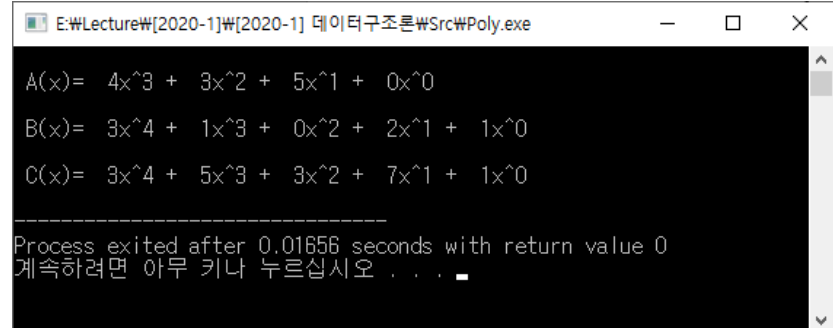
- 다항식 A, B가 아래와 같이 주어졌을 때 덧셈 연산 구현

$$A(x) = 4x^3 + 3x^2 + 5x$$

$$B(x) = 3x^4 + x^3 + 2x + 1$$

```
int main() {  
    polynomial A = {3, {4, 3, 5, 0}};  
    polynomial B = {4, {3, 1, 0, 2, 1}};  
  
    polynomial C = addPoly(A, B);  
  
    printf("\nA(x) = "); printPoly(A);  
    printf("\nB(x) = "); printPoly(B);  
    printf("\nC(x) = "); printPoly(C);  
  
    return 0;  
}
```

실행결과



```
E:\Lecture\2020-1\2020-1\데이터구조론\Src\Poly.exe  
A(x)= 4x^3 + 3x^2 + 5x^1 + 0x^0  
B(x)= 3x^4 + 1x^3 + 0x^2 + 2x^1 + 1x^0  
C(x)= 3x^4 + 5x^3 + 3x^2 + 7x^1 + 1x^0  
-----  
Process exited after 0.01656 seconds with return value 0  
계속하려면 아무 키나 누르십시오 . . .
```

- 희소행렬 A가 p.35처럼 주어졌을 때 전치연산 구현

```
int main() {
    int i, j;
    matrix a[11] = {
        {8,7,10}, {0,2,2}, {0,6,12},
        {1,4,7}, {2,0,23}, {3,3,31},
        {4,1,14}, {4,5,25}, {5,6,6},
        {6,0,52}, {7,4,11} };

    matrix b[11] = {0, };
    transposeSM(a, b);

    printf("Matrix a\n");
    for(i=0; i<11; i++)
        printf("%d: %d %d %d\n", i, a[i].row, a[i].col, a[i].value);
    printf("Transpose Matrix b\n");
    for(i=0; i<11; i++)
        printf("%d: %d %d %d\n", i, b[i].row, b[i].col, b[i].value);

    return 0;
}
```

실행결과

```
Matrix a
0: 8 7 10
1: 0 2 2
2: 0 6 12
3: 1 4 7
4: 2 0 23
5: 3 3 31
6: 4 1 14
7: 4 5 25
8: 5 6 6
9: 6 0 52
10: 7 4 11
Transpose Matrix b
0: 7 8 10
1: 0 2 23
2: 0 6 52
3: 1 4 14
4: 2 0 2
5: 3 3 31
6: 4 1 7
7: 4 7 11
8: 5 4 25
9: 6 0 12
10: 6 5 6
-----
Process exited after 0.01678 seconds with return value 0
계속하려면 아무 키나 누르십시오 . . .
```