

# 자료구조응용

## 10. Equivalence Class, Doubly Linked Circular List

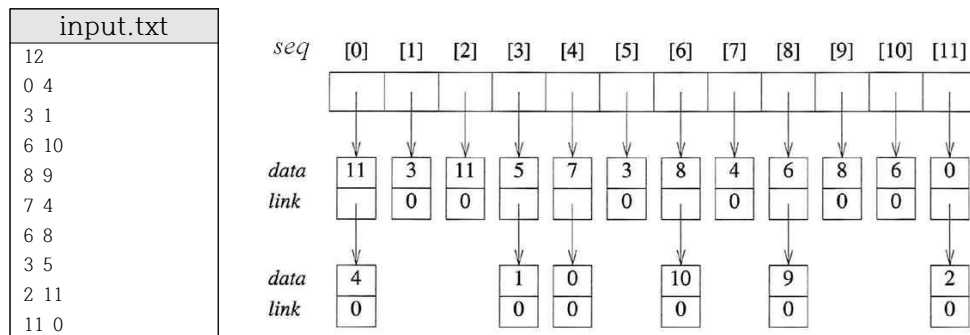
1. 다음과 같이 집합 S에 대한 동치관계(equivalence relation,  $\equiv$ )가 성립할 때 S의 동치류(equivalence class)를 구하는 프로그램을 작성하라. 연결리스트를 이용하여야 한다. 출력은 실행 예와 같아야 한다.

$S = \{ 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11 \}$

$0 \equiv 4, 3 \equiv 1, 6 \equiv 10, 8 \equiv 9, 7 \equiv 4, 6 \equiv 8, 3 \equiv 5, 2 \equiv 11, 11 \equiv 0$

### (1) 실행 순서

① 파일입력을 통해 집합 S의 크기와 동치관계를 나타내는 순서쌍 데이터를 입력받으면서 자료구조를 생성한다. ※ Program 4.22에서 키보드입력을 파일입력으로 수정하여 구현함



② 동치류를 구하여 출력한다.

### (2) 구현세부사항

program 4.22 참조

### (3) 실행 예

```

C:\WINDOWS\system32\cmd.exe
current size of S : 12
S = { 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11 }
input pairs : 0R4 3R1 6R10 8R9 7R4 6R8 3R5 2R11 11R0

New class:  0  11  4  7  2
New class:  1  3  5
New class:  6  8  10  9
계속하려면 아무 키나 누르십시오 . . .
    
```

2. 다음과 같이 정수 데이터를 입력하면서 “헤더노드를 가진 이중연결환형리스트 (doubly linked circular list)”를 만들고 실행 예와 같이 수행되는 프로그램을 작성하라.(디버깅 화면 캡처 포함)

### (1) 실행 순서

① 입력파일(“input.txt”)로 부터 데이터를 입력받는 순서대로 이중환형연결리스트의 마지막 노드로 추가 되도록 한다.

50	80	30	20	19	90	30	55	77	30
99	45	55	89	91	10	20	66	38	59
22	55	88	22	66	29	50	95	78	83

② 순방향과 역방향으로 노드의 데이터를 출력한다. (forward & backward )

③ 성적이 50점 이하인 노드를 삭제한다.

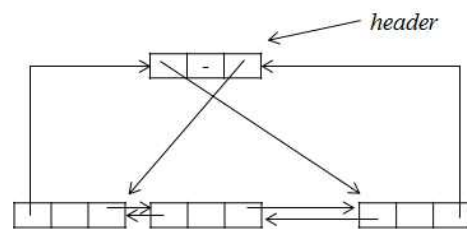
④ 순방향과 역방향으로 노드의 데이터를 출력한다.

⑤ 헤더를 제외한 모든 노드를 삭제한다.

### (2) 구현 세부사항

① 구조체 선언문

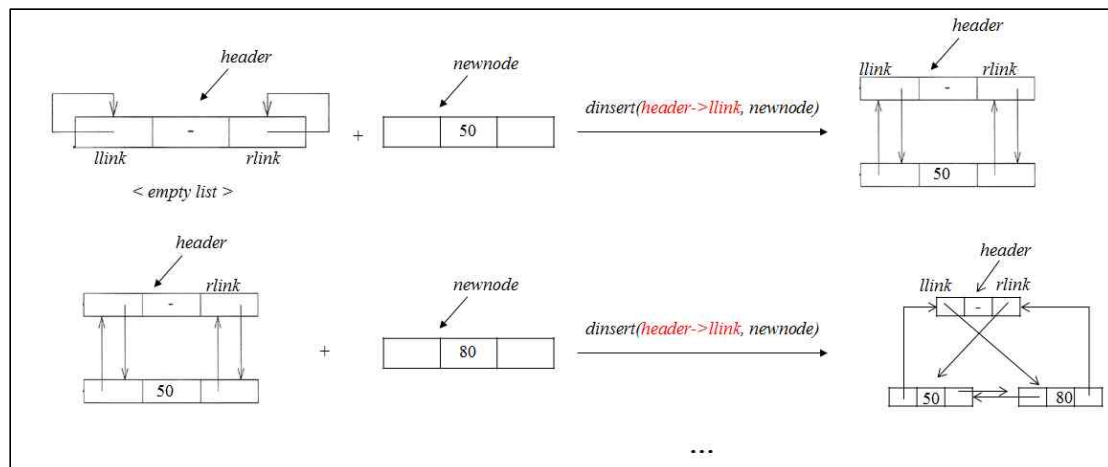
```
typedef struct node *nodePointer;
typedef struct node {
    nodePointer llink;
    int data;
    nodePointer rlink;
} node;
nodePointer header = NULL;
```



② 이중연결환형리스트의 Last node로 추가하기

- empty list를 생성한 후 새로운 노드를 하나씩 추가해 간다.

- header node의 llink는 last node pointer 역할을 한다. (\* dinsert 함수 호출)



### ③ 함수정의

```
void dinser(nodePointer node, nodePointer newnode)
{/* insert newnode to the right of node */
    newnode→llink = node;
    newnode→rlink = node→rlink;
    node→rlink→llink = newnode;
    node→rlink = newnode;
}
```

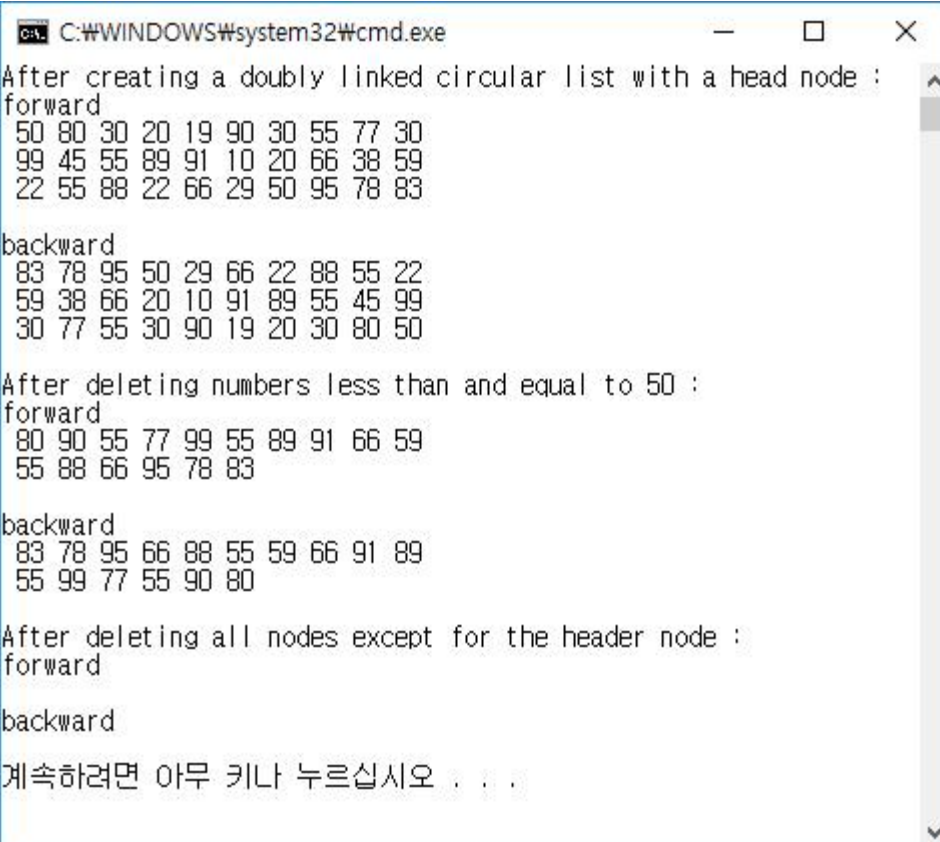
#### Program 4.26: Insertion into a doubly linked circular list

```
void ddelete(nodePointer node, nodePointer deleted)
{/* delete from the doubly linked list */
    if (node == deleted)
        printf("Deletion of header node not permitted.\n");
    else {
        deleted→llink→rlink = deleted→rlink;
        deleted→rlink→llink = deleted→llink;
        free(deleted);
    }
}
```

#### Program 4.27: Deletion from a doubly linked circular list

- 기타 함수들을 적절하게 선언하고 사용할 것(자료구조생성, 리스트 출력, 삭제관련함수 등)

### (3) 디버깅 및 실행 예



```
C:\WINDOWS\system32\cmd.exe
After creating a doubly linked circular list with a head node :
forward
50 80 30 20 19 90 30 55 77 30
99 45 55 89 91 10 20 66 38 59
22 55 88 22 66 29 50 95 78 83

backward
83 78 95 50 29 66 22 88 55 22
59 38 66 20 10 91 89 55 45 99
30 77 55 30 90 19 20 30 80 50

After deleting numbers less than and equal to 50 :
forward
80 90 55 77 99 55 89 91 66 59
55 88 66 95 78 83

backward
83 78 95 66 88 55 59 66 91 89
55 99 77 55 90 80

After deleting all nodes except for the header node :
forward

backward
계속하려면 아무 키나 누르십시오 . . .
```

이름	값	형식
backward	backward (1)	direction
forward	forward (0)	direction
header	0x00e38d40 {llink=0x00e441c0 {llink=0x00e44770 {llink=0x00e44508 {llink=0x00e44118 {llink=0x00e447e0 {...	node *
llink	0x00e441c0 {llink=0x00e44770 {llink=0x00e44508 {llink=0x00e44118 {llink=0x00e447e0 {llink=0x00e44738 {...	node *
data	-842150451	int
rlink	0x00e389f8 {llink=0x00e38d40 {llink=0x00e441c0 {llink=0x00e44770 {llink=0x00e44508 {llink=0x00e44118 {...	node *
llink	0x00e38d40 {llink=0x00e441c0 {llink=0x00e44770 {llink=0x00e44508 {llink=0x00e44118 {llink=0x00e447e0 {...	node *
data	50	int
rlink	0x00e38bb8 {llink=0x00e389f8 {llink=0x00e38d40 {llink=0x00e441c0 {llink=0x00e44770 {llink=0x00e44508 {...	node *
llink	0x00e389f8 {llink=0x00e38d40 {llink=0x00e441c0 {llink=0x00e44770 {llink=0x00e44508 {llink=0x00e44118 {...	node *
data	80	int
rlink	0x00e38a30 {llink=0x00e38bb8 {llink=0x00e389f8 {llink=0x00e38d40 {llink=0x00e441c0 {llink=0x00e44770 {...	node *
llink	0x00e38bb8 {llink=0x00e389f8 {llink=0x00e38d40 {llink=0x00e441c0 {llink=0x00e44770 {llink=0x00e44508 {...	node *
data	30	int
rlink	0x00e38bf0 {llink=0x00e38a30 {llink=0x00e38bb8 {llink=0x00e389f8 {llink=0x00e38d40 {llink=0x00e441c0 {...	node *
llink	0x00e38a30 {llink=0x00e38bb8 {llink=0x00e389f8 {llink=0x00e38d40 {llink=0x00e441c0 {llink=0x00e44770 {...	node *
data	20	int
rlink	0x00e38c28 {llink=0x00e38bf0 {llink=0x00e38a30 {llink=0x00e38bb8 {llink=0x00e389f8 {llink=0x00e38d40 {...	node *
llink	0x00e38bf0 {llink=0x00e38a30 {llink=0x00e38bb8 {llink=0x00e389f8 {llink=0x00e38d40 {llink=0x00e441c0 {...	node *
data	19	int
rlink	0x00e38cd0 {llink=0x00e38c28 {llink=0x00e38bf0 {llink=0x00e38a30 {llink=0x00e38bb8 {llink=0x00e389f8 {...	node *
llink	0x00e38c28 {llink=0x00e38bf0 {llink=0x00e38a30 {llink=0x00e38bb8 {llink=0x00e389f8 {llink=0x00e38d40 {...	node *
data	90	int
rlink	0x00e38d08 {llink=0x00e38cd0 {llink=0x00e38c28 {llink=0x00e38bf0 {llink=0x00e38a30 {llink=0x00e38bb8 {...	node *
llink	0x00e38cd0 {llink=0x00e38c28 {llink=0x00e38bf0 {llink=0x00e38a30 {llink=0x00e38bb8 {llink=0x00e389f8 {...	node *
data	30	int
rlink	0x00e38d78 {llink=0x00e38d08 {llink=0x00e38cd0 {llink=0x00e38c28 {llink=0x00e38bf0 {llink=0x00e38a30 {...	node *

## ■ 제출 형식

- 공학인증 시스템(ABEEK)에 과제를 올릴 때 제목:
  - 1차 제출: 학번\_이름\_DS\_10(1), 2차 제출: 학번\_이름\_DS\_10(2)
- 솔루션 이름 : DS\_10
- 프로젝트 이름 : 1, 2
- 실행화면을 캡처하여 한글파일에 추가 후 솔루션 폴더에 포함.
- 한글 파일명 : 학번\_이름.hwp
- 솔루션 폴더를 압축하여 제출할 것.
- 솔루션 압축 파일 명:
  - 1차 제출: 학번\_이름\_DS\_10(1).zip, 2차 제출: 학번\_이름\_DS\_10(2).zip
- 제출은 2회 걸쳐 가능(수정 시간 기준으로 처리)