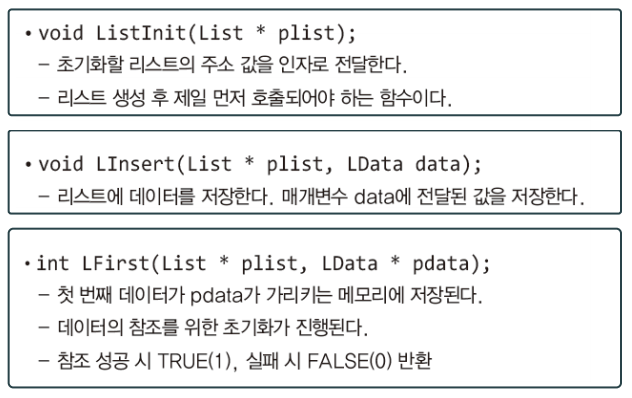
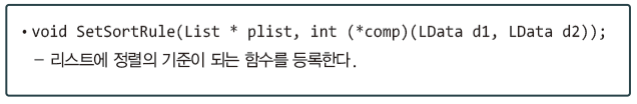
**연결 리스트의 ADT 정의**



LData는 저장 대상의 자료형을 결정할 수 있도록 typedef로 선언된 자료형의 이름이다.



**새로 추가된 함수**



**새 노드의 추가 위치에 따른 장점과 단점**

새 노드를 연결 리스트의 머리에 추가하는 경우

∙ 장점 포인터 변수 tail이 불필요하다.

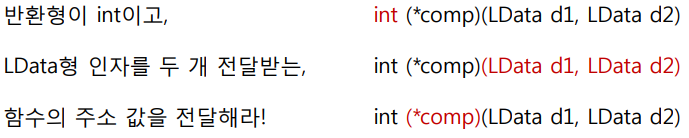
∙ 단점 저장된 순서를 유지하지 않는다.

새 노드를 연결 리스트의 꼬리에 추가하는 경우

∙ 장점 저장된 순서가 유지된다.

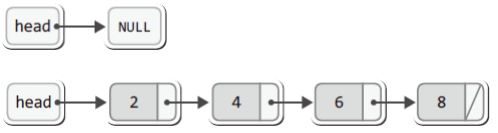
∙ 단점 포인터 변수 tail이 필요하다.

**정렬의 기준을 설정하기 위해 정의된 SetSortRule함수**

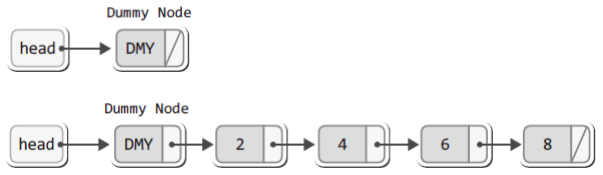


**더미 노드 기반 연결 리스트**

더미 노드가 없는 경우



더미 노드가 있는 경우

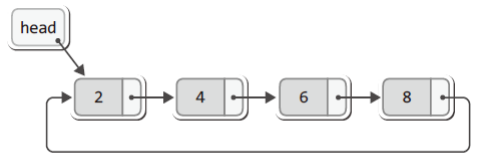


**원형 연결 리스트**

단순 열결 리스트

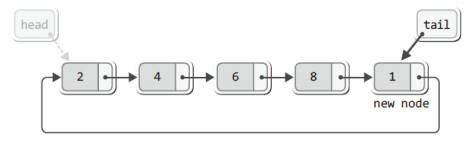


원형 연결 리스트



**원형 연결 리스트의 대표적인 장점**

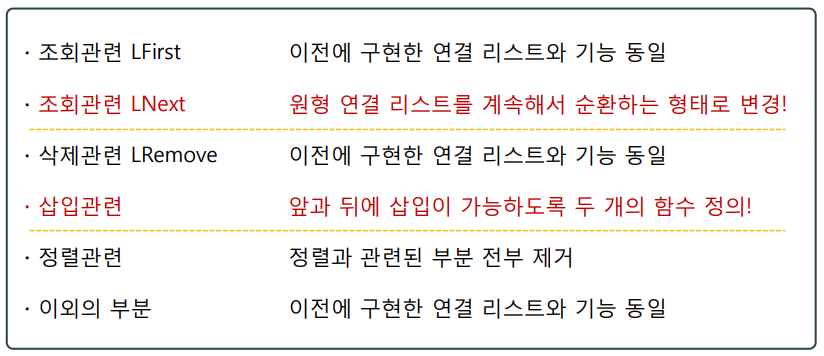
단순 연결 리스트처럼 머리와 꼬리를 가리키는 포인터 변수를 각각 두지 않아도, 하나의 포인터 변수만 있어도 머리 또는 꼬리에 노드를 간단히 추가할 수 있다.



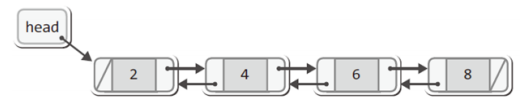
앞서 소개한 모델을 기반으로는 위의 장 점을 살리기 어렵다. 그래서 우리는 변형된, 그러나 보다 일반적이라고 인식이 되고 있는 변경된 원형 연결 리스트를 구현한다

∙ 꼬리를 가리키는 포인터 변수는? tail

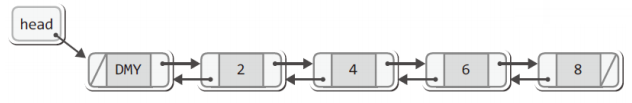
∙ 머리를 가리키는 포인터 변수는? tail->next



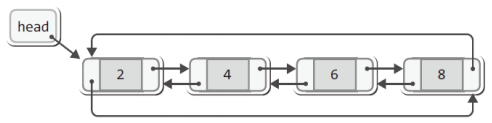
**양방향 연결 리스트**



일반적인 양방향 연결 리스트

****

더미 노드 기반 양방향 연결 리스트



원형 기반 양방향 연결 리스트