**Linked list**

연결 리스트(Linked list)는 각 노드가 데이터와 포인터를 가지고 한 줄로 **연결**되어 있는 방식으로 데이터를 저장하는 자료 구조이다. 이름에서 말하듯이 데이터를 담고 있는 노드들이 **연결**되어 있는데, 노드의 포인터가 다음이나 이전의 노드와의 **연결**을 담당하게 된다.

연결 리스트는 배열과 같이 크기가 정해져 있는 자료구조와 달리 메모리를 동적으로 사용할 수 있는 장점이 있다. 반면 다음 노드를 가리키는 포인터를 저장할 메모리를 사용해야 하기 때문에, 정적인 메모리 사용에 비해 메모리가 추가적으로 필요하다.

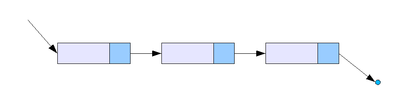
참고사이트: <https://ko.wikipedia.org/wiki/%EC%97%B0%EA%B2%B0_%EB%A6%AC%EC%8A%A4%ED%8A%B8\>

<https://imtaeheewon.tistory.com/14>

**종류**

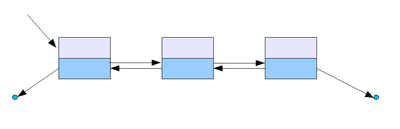
연결 리스트의 종류로는 단일 연결 리스트, 이중 연결 리스트 등이 있다.

* **단일 연결 리스트**



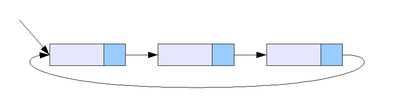
단일 연결 리스트는 각 노드에 자료 공간과 한 개의 포인터 공간이 있고, 각 노드의 포인터는 다음 노드를 가리킨다.

* **이중 연결 리스트**



이중 연결 리스트의 구조는 단일 연결 리스트와 비슷하지만, 포인터 공간이 두 개가 있고 각각의 포인터는 앞의 노드와 뒤의 노드를 가리킨다.

* **원형 연결 리스트**



원형 연결 리스트는 일반적인 연결 리스트에 마지막 노드와 처음 노드를 연결시켜 원형으로 만든 구조이다.

**코드 구현 – in C**

struct NODE { // 연결 리스트의 노드 구조체

struct NODE \*next; // 다음 노드의 주소를 저장할 포인터

int data; // 데이터를 저장할 멤버

};

int main()

{

struct NODE \*head = malloc(sizeof(struct NODE)); // 머리 노드 생성

// 머리 노드는 데이터를 저장하지 않음

struct NODE \*node1 = malloc(sizeof(struct NODE)); // 첫 번째 노드 생성

head->next = node1; // 머리 노드 다음은 첫 번째 노드

node1->data = 10; // 첫 번째 노드에 10 저장

struct NODE \*node2 = malloc(sizeof(struct NODE)); // 두 번째 노드 생성

node1->next = node2; // 첫 번째 노드 다음은 두 번째 노드

node2->data = 20; // 두 번째 노드에 20 저장

node2->next = NULL; // 두 번째 노드 다음은 노드가 없음(NULL)

struct NODE \*curr = head->next; // 연결 리스트 순회용 포인터에 첫 번째 노드의 주소 저장

while (curr != NULL) // 포인터가 NULL이 아닐 때 계속 반복

{

printf("%d\n", curr->data); // 현재 노드의 데이터 출력

curr = curr->next; // 포인터에 다음 노드의 주소 저장

}

free(node2); // 노드 메모리 해제

free(node1); // 노드 메모리 해제

free(head); // 머리 노드 메모리 해제

return 0;

}

코드 출처 : <https://dojang.io/mod/page/view.php?id=645>

**연결 리스트에서 노드를 추가하는 규칙:**

1. 노드에 메모리 할당
2. next 멤버에 다음 노드의 메모리 주소 저장
3. data 멤버에 데이터 저장
4. if - 마지막 노드라면 next 멤버에 NULL 저장

**이중? 원형?**

이중 연결 리스트와 원형 연결 리스트는 메모리의 사용에서 약간의 차이가 있을 뿐, 기본적인 구조차이는 없다. 하지만 사용의 차이로 인해 리스트에 노드를 추가할 때에 효율이 차이가 난다.

원형 연결 리스트는 리스트의 첫번째 노드의 앞 노드를 가리키는 포인터가 이미 리스트의 마지막 노드를 가리키고 있으므로 이를 찾는 연산이 필요 없다는 특징이 있다. 원형 연결 리스트는 구현하는 데에 있어 조금 어려워진다는 단점 이외에는 리스트에 노드를 추가할 때에 효율적이다.

출처 : <https://imtaeheewon.tistory.com/14>