고급C프로그래밍

12 양방향 연결 리스트

코딩테스트 공지

- **일시/장소**
  - ✓ 12/5일(수) 13:00, 303호

- **진행 방안**
  - ✓ 13:00시에 조교 지시에 따라 입실
  - ✓ 입실 후 화면에 표시된 자리 배치도를 따라 자리에 착석
  - ✓ 구현 후, 조교의 지시에 따라 결과물 제출
  - ✓ 일체 전자기기 사용 시 퇴실, F학점 처리

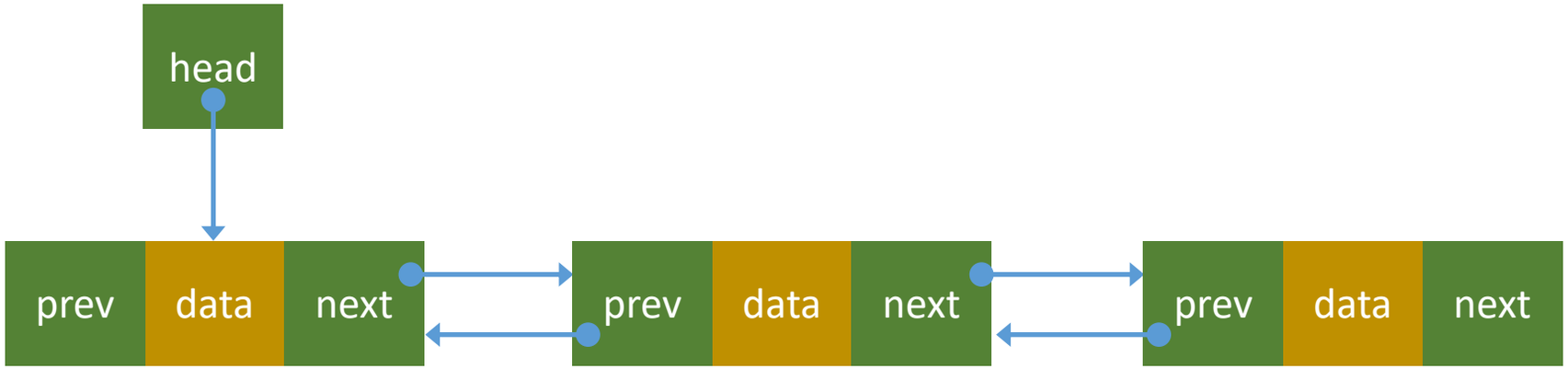
기말고사 공지

- **일시/장소**
  - ✓ 12/13일(토) 13:00, 장소 추후 공지

- **시험 범위**
  - ✓ 양방향 연결 리스트까지 배운 모든 내용

이중 연결 리스트 (양방향 연결 리스트) (1/7)

- **앞, 뒤 노드와 연결을 유지**
  - ✓ 기존 단일 연결 리스트에서 앞 노드와의 연결을 추가
  - ✓ 노드 탐색 및 리스트 정렬 등에서 단일 연결 리스트 대비 장점을 보유



이중 연결 리스트 (양방향 연결 리스트) (2/7)

<1단계> 노드 정의

✓ 노드 구성요소 : 데이터와 이전, 다음 노드를 가리키는 포인터

```
struct Node {
        int data;
        struct Node* prev;      //이전 노드를 가리키는 포인터
        struct Node* next;      //다음 노드를 가리키는 포인터
};
```

이중 연결 리스트 (양방향 연결 리스트) (3/7)

<2단계> 새로운 노드 만들기

```
//새로운 노드에 저장할 데이터를 매개변수로 전달
//만들어진 노드의 주소를 반환
struct Node* create_node(int data) {

        //동적 메모리 할당으로 노드를 생성
        struct Node* new_node = (struct Node*)malloc(sizeof(struct Node));

        //매개변수로 전달받은 데이터를 노드에 저장
        new_node->data = data;

        //이전, 다음 노드를 가리킬 포인터는 NULL로 초기화
        new_node->prev = NULL;
        new_node->next = NULL;

        return new_node;
}
```

이중 연결 리스트 (양방향 연결 리스트) (4/7)

<3단계> 연결 리스트에 노드를 추가하기

```
//head는 연결 리스트의 시작 노드를 가리킴
Struct Node* insert_node(struct Node* head, int data) {
        //연결 리스트에 추가할 노드를 생성
        struct Node* new_node = create_node(data);

        //기존 연결 리스트에 노드가 하나도 없을 경우
        if (head == NULL) { head = new_node; }

        //마지막 노드를 찾아서 새로 만든 노드를 추가
        else {
                struct Node* temp = head;
                while (temp->next != NULL) { temp = temp->next; }
                temp->next = new_node;
                new_node->prev = temp;
        }  return head; }
```

이중 연결 리스트 (양방향 연결 리스트) (5/7)

<4단계> 연결 리스트에 노드를 삭제하기

```
struct Node* delete_node(struct Node* head, int target) {

//노드 삭제를 위해서는 1개의 노드 포인터가 필요
        struct Node* current = head;
        struct Node* prev = NULL;

// 첫 번째 노드가 삭제 대상인 경우 head를 다음 노드로 변경하고 노드를 삭제
        if (current != NULL && current->data == target) {
                head = current->next;
                head->prev = NULL;
                free(current);
                return head; }
```

이중 연결 리스트 (양방향 연결 리스트) (6/7)

<4단계> 연결 리스트에 노드를 삭제하기 (계속)

```
// 삭제 대상 노드를 찾을 때 까지 노드를 건너가며 계속 탐색
        while (current != NULL && current->data != target) {
                prev = current;
                current = current->next;  }

//삭제 대상 노드가 없을 경우
        if (current == NULL) {
                printf("삭제할 노드가 없습니다.\n");
                return head; }

// 삭제 대상 노드가 마지막 노드일 경우
        if (current->next == NULL) {
                current->prev->next = NULL;
                free(current);
                return head; }
```

이중 연결 리스트 (양방향 연결 리스트) (7/7)

<4단계> 연결 리스트에 노드를 삭제하기 (계속)

```
// 삭제 대상 노드가 중간에 위치할 경우
        current->prev->next = current->next;
        current->next->prev = current->prev;
        free(current);
        return head; }
```

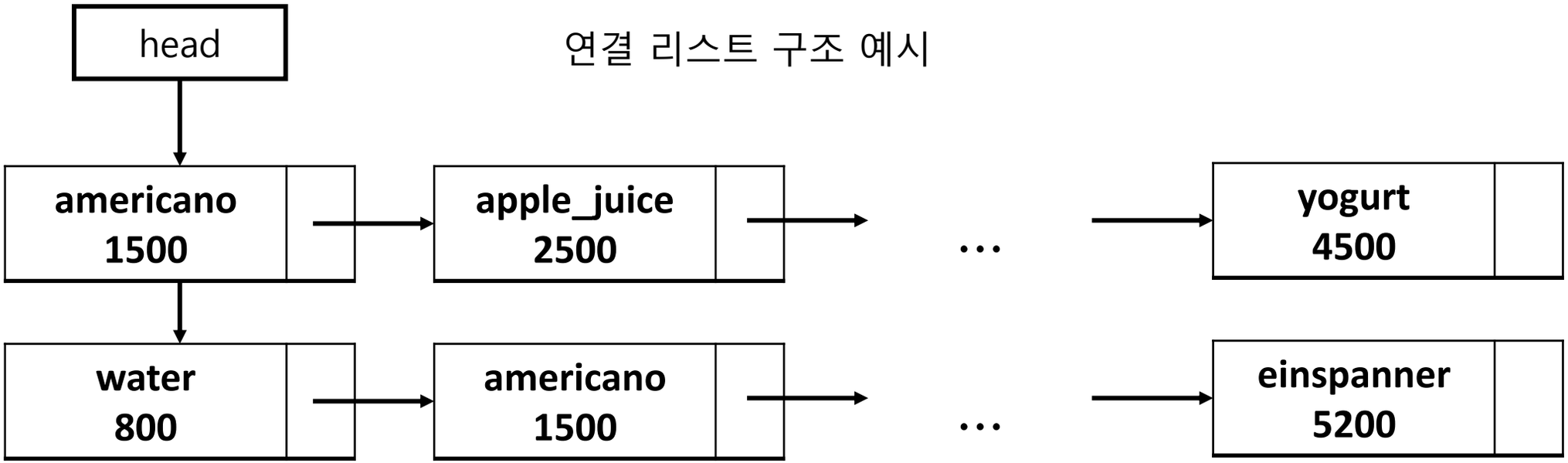
실습

[실습 1] 간단한 양방향 연결 리스트 생성 프로그램 확장

- **11주차 [실습 1~2]의 프로그램을 양방향 연결 리스트를 사용하여 재구현**

[실습 2] 음료수 이름/가격순 정렬

- **이름순 정렬/가격순 정렬 연결 리스트를 생성**
  - ✓ drink.txt에서 이름과 가격 정보를 읽어옴
  - ✓ 이름순 정렬 결과와 가격순 정렬 정보를 출력(음료 이름과 가격을 출력)



Q & A