9주차 결과보고서

전공: 컴퓨터공학과 학년: 4학년 학번: 20212022 이름: 이예준

**1.**

텍스트, 폰트, 스크린샷이(가) 표시된 사진

자동 생성된 설명

Linked List를 이용하여 Ranking 시스템을 구현하였다.

위의 코드는 Linked List에 사용하는 Node 구조체이며 점수, 이름 그리고 다음 Node를

가리키는 포인터로 구성되어 있다.

텍스트, 스크린샷, 소프트웨어이(가) 표시된 사진

자동 생성된 설명

createRankList() 함수는 파일을 불러와서 파일에 있는 정보들을 Linked List 형식으로

저장하는 함수이다. 먼저 fopen으로 파일을 연 뒤 fscanf로 랭킹의 정보 수를 rank\_num에

저장한다. 그 다음 rank\_num만큼 for문을 돌려준다. fscanf로 파일에서 이름과 점수를 가져와

각각 rank\_name과 rank\_score에 저장한다. 그 다음 동적 할당으로 노드 newNode를 생성 후

rank\_name과 rank\_score를 newNode에 저장한다.

만약 노드를 처음에 만든 것이라면 Linked List에서 첫 노드를 가리키는 포인터 head가

newNode를 가리키도록 한다. 그것이 아니라면 포인터 list를 선언하여 head를 가리키도록 한 뒤

while문을 이용하여 Linked List의 맨 끝까지 이동하고, 그 끝에 newNode를 연결한다.

이렇게 연결하는 이유는 파일의 정보가 오름차순으로 저장되어 있어

Linked List도 오름차 순으로 연결하기 위함이다. 이렇게 연결해 놓으면 나중에

Linked List 정보를 파일에 저장할 때, 오름차순으로 저장하기 쉽다.

시간 복잡도 : while문을 통해 매번 Linked List의 끝으로 이동해야 되기 때문에(1, 2, 3,.., N)

시간 복잡도는 O(n2) 이다.

공간 복잡도 : 파일의 정보 수만큼 노드를 만들어야 되기 때문에 시간 복잡도는 O(n) 이다.

텍스트, 스크린샷, 폰트이(가) 표시된 사진

자동 생성된 설명

텍스트, 스크린샷, 소프트웨어, 디스플레이이(가) 표시된 사진

자동 생성된 설명

rank() 함수는 Linked List를 이용하여 사용자의 의도에 따라 Linked List의 정보를 탐색 및

삭제하는 함수이다. 실습에서는 ‘1. List ranks from X to Y’ 를 구현했다.

X와 Y를 입력 받으면 그 범위에 해당하는 랭킹 정보를 출력하는 함수이다.

처음에는 X에 1, Y에 rank\_num을 저장하고, list 포인터는 head를 가리킨다.

scanw를 통해 사용자의 입력으로 X값을 바꾸게 되는데 그냥 enter키를 치게 되는 경우에는

값이 변하지 않고 초기에 설정한 값을 유지한다.

X, Y값을 입력 받았을 때, X가 Y보다 크게 되면 유효하지 않은 범위 설정이기 때문에

에러 메세지를 출력하게 된다.

정상적으로 범위가 정해졌다면 for문을 이용하여 Linked List를 한바퀴 순회한다.

이때, 초기에 int count = 1를 선언한 뒤 for문을 한바퀴 돌때마다 count 값이 1씩 증가하는데

이 count값이 X ≤ count ≤ Y 라면 출력 대상이기 때문에 그 시점의 Linked List 정보를 출력한다.

시간 복잡도 : Linked List를 한바퀴 순회하기 때문에 시간 복잡도는 O(n) 이다.

공간 복잡도 : 추가적인 메모리 사용이 없기 때문에 공간 복잡도는 O(1) 이다.

텍스트, 스크린샷, 폰트이(가) 표시된 사진

자동 생성된 설명

writeRankFile() 함수는 Linked List의 정보를 파일에 저장하는 함수이다.

먼저 fprintf를 이용해서 Linked List의 정보 수 rank\_num을 파일에 저장하고,

Linked List를 한바퀴 순회하면서 모든 정보를 파일에 저장한다.

시간 복잡도 : Linked List를 한바퀴 순회하기 때문에 시간 복잡도는 O(n) 이다.

공간 복잡도 : 추가적인 메모리 사용이 없기 때문에 공간 복잡도는 O(1) 이다.

텍스트, 스크린샷, 폰트, 소프트웨어이(가) 표시된 사진

자동 생성된 설명

텍스트, 스크린샷, 소프트웨어이(가) 표시된 사진

자동 생성된 설명

newRank() 함수는 게임 종료 시 종료된 시점의 점수와 사용자가 입력한 이름을

Linked List의 적절한 위치에 저장하는 함수이다.

먼저 이름을 입력 받아서 str에 저장한다. 그 다음 새로운 노드 newNode를 동적 할당하여

이름 str과 점수 score를 노드에 저장한다.

만약 Linked List에 아무런 정보가 없다면 head가 newNode를 가리키게 한다.

아니라면 cur포인터와 prev포인터를 선언하고 cur 포인터가 head를 가리키게 한다.

newNode의 점수가 만약 첫 노드의 점수보다 크다면 newNode의 next가 head를 가리키게 하고,

head 포인터를 newNode로 이동시킨다. 아니라면 cur이 가리키는 노드의 점수가

newNode의 점수보다 작거나 List의 끝까지 갈 때까지 while문을 실행하며 한번 실행될 때마다

prev가 cur을 가리키고, cur은 다음 노드로 이동한다.

while문이 끝났을 때, List의 끝에 있다면 끝 노드 다음에 newNode를 연결하며 그게 아니라면

멈춘 위치에 newNode를 연결한다. 함수 마지막에 rank\_num에 1을 더한다.

시간 복잡도 : 최악의 경우 Linked List 끝까지 이동해야 되기 때문에 시간 복잡도는 O(n) 이다.

공간 복잡도 : newNode, cur, prev를 한번씩만 동적 할당하기 때문에 공간 복잡도는 O(1) 이다.

**2.**

Linked List에 새로운 노드를 삽입할 때, 삽입하는 위치에 따라 다른 알고리즘을

사용해야 하는 것에 대해서 이번 실습에서 더 깊이 이해하게 됐다.

1. 맨 앞에 삽입할 때 : 새로운 노드는 기존에 있는 List의 첫 노드를 가리키고, List의 첫 노드를

가리키는 포인터 head는 새로운 노드를 가리키도록 수정한다.

2. 중간에 삽입할 때 : while문에서 나오게 되면 포인터 prev과 cur사이에 노드를 삽입해야 한다.

따라서 새로운 노드의 next는 cur을 가리키고, prev의 next는

새로운 노드를 가리키도록 해야 한다.

3. 맨 끝에 삽입할 때 : while문에서 나오게 되면 포인터 cur은 NULL을 가리키고

prev는 마지막 노드를 가리키고 있기 때문에 prev의 next가

새로운 노드를 가리키도록 해줘야 한다.

또한 과제를 통해 구현한 노드 삭제 알고리즘도 삭제하는 위치에 따라 알고리즘이 달라진다.

1. 맨 앞에서 삭제할 때

: prev 포인터로 List의 첫 노드를 가리키고, List의 첫 노드를 가리키는 포인터 head는

그 다음 노드를 가리키도록 수정한다.

2. 그 외의 위치에서 삭제할 때

: while문을 통해서 List를 순회하다가 목표하는 위치에 도달하면 list 포인터는 삭제해야 하는

노드를 가리키고 있고, prev 포인터는 바로 전 노드를 가리키고 있다.

prev가 가리키는 노드의 next가 list가 가리키고 있는 노드의 next 노드를 가리키도록 수정하고,

prev 포인트는 list 포인터가 가리키는 노드를 가리키도록 한다.

*\*이 과정을 마친 후 prev 포인트는 free 명령어를 통해 동적 해제를 해준다.*

*텍스트, 스크린샷, 소프트웨어이(가) 표시된 사진

자동 생성된 설명*