202155655 정진택

1. 학습 내용

1.1 그래프 탐색 (DFS, BFS)

* 학습 방법  
  그래프 탐색 알고리즘인 \*\*DFS(깊이 우선 탐색)\*\*와 \*\*BFS(너비 우선 탐색)\*\*를 먼저 이론적으로 학습한 후, 각각의 알고리즘을 구현하고, 백준 문제에서 이를 적용해 풀이했습니다.
* 느낀 점  
  DFS와 BFS는 기본적인 탐색 알고리즘이지만, 각각의 활용도와 구현 방법이 다르기 때문에 처음에는 어려움을 겪었습니다. 특히 BFS는 큐를 사용하여 순차적으로 탐색하므로, 최단 경로 문제에서 유용하다는 점을 깨달았습니다.  
  또한, DFS는 스택을 사용하여 깊이를 우선적으로 탐색하기 때문에 그래프의 모든 경로를 빠짐없이 탐색하는 데 적합함을 느꼈습니다.
* 향후 개선 사항  
  DFS와 BFS를 더 다양한 문제에 적용하여 이해도를 높이고, 탐색 효율성을 고려한 최적화 방법을 고민할 필요가 있습니다. 또한, 그래프 문제에서 다익스트라나 플로이드-워셜과 같은 더 복잡한 알고리즘을 연계해 훈련할 계획입니다.

1.2 동적 계획법 (DP)

* 학습 방법  
  동적 계획법(DP)을 처음 접했을 때는 상태 전이의 개념이 낯설었으나, 이를 점화식으로 풀어보면서 점차 익숙해졌습니다.
  + 11053번 (가장 긴 증가하는 부분 수열) 문제에서는 \*\*O(N^2)\*\*의 시간 복잡도를 가진 이중 루프 방식을 사용한 후, \*\*O(N log N)\*\*으로 최적화된 이분 탐색을 적용하여 풀었습니다.
* 느낀 점  
  DP는 문제를 푸는 데 있어서 큰 그림을 그리기 어렵다는 점에서 처음엔 난이도가 높았지만, 재귀적 문제 해결을 이해하고 메모이제이션과 타뷸레이션 기법을 적용하면서 점차 해결할 수 있었습니다.  
  특히 배낭 문제는 DP의 전형적인 예제로, 반복되는 계산을 줄이는 기법이 얼마나 중요한지 느꼈습니다. DP 문제는 문제를 작은 단위로 분할하여 해결하는 방식이므로, 이를 최적화하는 것이 중요하다는 점을 깨달았습니다.
* 향후 개선 사항  
  더 많은 동적 계획법 문제를 풀고, 문제에서 요구하는 최적화 방식을 찾는 연습이 필요합니다. 특히 중복 계산을 피하는 방법이나 상태 공간을 줄이는 기법에 대한 이해를 더욱 깊게 해야 합니다.

1.3 그리디 알고리즘

* 학습 방법  
  그리디 알고리즘은 매번 최적이라고 생각되는 선택을 하는 방식으로, 이를 처음 배울 때는 최적해를 보장하는지에 대한 확신이 부족했습니다. 그러나 여러 문제를 풀어보면서, 그리디 알고리즘이 문제 조건에 맞는 경우 매우 효율적이고 직관적인 방식임을 알게 되었습니다.
  + 11399번 (ATM) 문제는 그리디 알고리즘을 활용하여, 시간을 최소화하는 방법을 배웠습니다. 각 사람의 ATM 사용 시간이 짧은 순서대로 처리하는 방식입니다.
  + 1931번 (회의실 배정) 문제는 회의실을 효율적으로 배치하기 위한 그리디 알고리즘으로, 종료 시간이 빠른 회의부터 배치하여 회의실을 최대한 활용하는 방식입니다.
* 느낀 점  
  그리디 알고리즘은 직관적이고 코드 구현이 간단하지만, 해가 최적임을 보장할 수 있는 조건을 정확히 파악해야 한다는 점에서 어려움을 겪었습니다. 여러 문제를 풀면서 최적 조건을 찾는 눈을 키울 수 있었습니다.
* 향후 개선 사항  
  그리디 알고리즘은 항상 최적해를 보장하는지 검토해야 하므로, 조건을 정확히 파악하는 연습을 더 해야 합니다. 다양한 그리디 알고리즘 문제를 풀어보며 문제 해결 패턴을 익히고, 그리디 알고리즘을 적절히 적용하는 능력을 키워야 합니다.

1.4 이분 탐색

* 학습 방법  
  이분 탐색은 정렬된 데이터에서 특정 조건을 만족하는 값을 찾는 알고리즘입니다. 이를 처음에는 단순히 정렬된 리스트에서의 탐색만으로 이해했지만, 문제를 풀어가면서 응용 방식이 중요하다는 것을 알게 되었습니다.
  + 2805번 (나무 자르기) 문제는 이분 탐색을 사용하여, 최소 높이를 찾는 문제를 해결했습니다.
* 느낀 점  
  이분 탐색은 문제를 해결하는 데 있어서 매우 빠르고 효율적인 방법이지만, 시작과 끝을 어떻게 설정할지에 대한 고민이 필요했습니다. 특히 이분 탐색의 경계 설정을 어떻게 할지에 대해 고민하면서 많은 도움을 얻었습니다.
* 향후 개선 사항  
  이분 탐색은 문제의 조건을 정확히 파악한 후, 적절한 경계 설정을 해야 하므로, 여러 이분 탐색 문제를 풀어 보며 경계 설정의 패턴을 익히는 것이 중요합니다.

2. 활동에서 느낀 점

* 다양한 접근법의 중요성  
  같은 문제를 풀이하더라도 사람마다 다른 접근법을 사용했는데, 이를 통해 더 효율적인 풀이 방법을 배우고 사고의 폭을 넓힐 수 있었습니다.
* 협업의 가치  
  어려운 문제를 해결할 때, 스터디원과의 의견 교환을 통해 새로운 통찰을 얻었습니다.
* 기본기 강화 필요성  
  초기에 놓쳤던 알고리즘 이론이나 기초적인 부분에서 종종 막히는 경우가 있었는데, 이는 기초를 탄탄히 다질 필요성을 느끼게 했습니다.

3. 향후 계획

* 심화 문제 풀이  
  기존에 다룬 알고리즘을 응용한 난이도 높은 문제 풀이 시도.
* 최적화에 집중  
  단순히 문제를 푸는 데 그치지 않고, 코드의 시간 복잡도와 공간 복잡도를 분석하고 개선.
* 코딩 테스트 준비  
  실전 대비를 위해 기업별 코딩 테스트 기출 문제 풀이.

4. 리드미 요약

* 스터디 목적  
  알고리즘 문제 풀이를 통해 컴퓨터 과학적 사고력을 기르고, 팀원들과의 협업을 통해 새로운 풀이 방법과 통찰을 얻기.
* 학습한 알고리즘 및 주요 문제

1. 그래프 탐색 (DFS, BFS)

* + DFS(깊이 우선 탐색)와 BFS(너비 우선 탐색)를 활용하여 그래프에서의 경로를 찾는 문제를 풀었습니다.

2. 동적 계획법 (DP)

* + 재귀적 접근을 통한 최적화 문제 해결 기법인 동적 계획법을 학습했습니다.
  + 문제 예시: 백준 11053번 (가장 긴 증가하는 부분 수열)

3. 그리디 알고리즘

* + 최적의 선택을 하여 문제를 해결하는 그리디 알고리즘을 통해 최적화된 해결 방법을 배웠습니다.
  + 문제 예시: 백준 11399번 (ATM), 1931번 (회의실 배정).

4. 이분 탐색

* + 정렬된 데이터에서 조건에 맞는 값을 빠르게 찾는 알고리즘인 이분 탐색을 활용한 문제 해결을 학습했습니다.
  + 문제 예시: 2805번 (나무 자르기).

5. 기타 알고리즘

* + 알고리즘 이론을 토대로 다양한 문제를 풀면서 알고리즘의 실용성과 효율성을 체득했습니다.
* 성과
  + 다익스트라, 플로이드-워셜 등 주요 알고리즘 완전 이해.
  + 시간 복잡도 및 최적화의 중요성 인식.
  + 협업을 통한 새로운 문제 접근 방식 발견.
* 향후 목표
  + 고난도 문제 풀이 능력 향상.
  + 실전 코딩 테스트 준비와 알고리즘 최적화 능력 강화.
  + 문제 풀이에서 더 많은 알고리즘을 융합하여 효율적인 해결 방법을 찾는 연습.