DB 2차 과제 보고서

12171726 황현택

연락처 010-6403-1867, [fasdgas05@gmail.com](mailto:fasdgas05@gmail.com)

1. 구현 및 설명

먼저 b+tree의 구현을 위하여 BPTree라는 클래스를 구현하였습니다. 이 클래스는 b+tree에서 작동하는 insert, find, range search를 지원하고 과제에서 요구한 level<0>와 level<1>의 key값들을 출력하는 print함수도 구현하였습니다. 그리고 BPTree 클래스 내부에 node, leafNode, nonLeafNode클래스를 구현하였습니다. 먼저 Node 클래스는 leafNode와 nonLeafnode의 base가 되는 클래스로 entry 배열의 정보를 저장할 포인터와 블록내 최대 entry개수의 정보를 담고 있습니다. leafNode와 nonLeafNode 클래스는 Node를 상속해 구현되었습니다. leafNode는 next\_bid라는 변수로 sibling의 bid를 저장하는 변수가 있습니다. 그리고 nonLeafNode는 least\_bid라는 변수로 맨 앞의 포인터를 저장하였습니다. 두 클래스 모두 파일에서 node를 읽어오는 생성자와 빈 노드를 만드는 생성자가 있습니다. 또 writeNode 라는 함수를 통해 update된 node의 내용을 파일에 기록할 수 있습니다. writeNode에서는 leaf인지 non leaf인지에 따라서 순서를 맞추어 파일에 작성하였습니다. 새로운 노드가 생성될 때는 BPTree 내부에 다음 사용할 수 있는 bid의 정보를 담고 있는 availBid라는 변수를 사용하고 업데이트 시켜주었습니다.

insert에서는 b+tree에서의 insert연산을 구현하였습니다. key 값을 통해 노드가 삽입될 위치를 찾았습니다. 탐색을 할 때는 depth를 이용하여 leaf node인지 non leaf node인지 구분하였습니다. 탐색 후 삽입될 leaf node에 entry를 삽입한 뒤 sort함수를 통해 정렬을 해주었습니다. 삽입 뒤 node내 entry수가 조건에 만족하는지 검사를 한 뒤 split이 필요하다면 현재 leaf node부터 root node까지 검사하며 split하였습니다. split이 필요할 때는 새로운 블록에 node를 저장하였습니다.

find에서는 key 값을 통해 해당 entry를 b+tree에서 찾는 연산을 수행하였습니다. 위의 insert와 동일한 방식으로 depth를 이용해 non leaf 와 leaf node를 구분해가며 해당 key를 찾을 수 있는 leaf node를 찾고 그 내부에서 탐색을 수행하여 해당 entry를 return하였습니다.

range Search에서는 lower와 upper를 인수로 받아 해당 범위안에 있는 모든 key값을 vector에 넣어 return 하였습니다. 먼저 find와 동일한 방식으로 lower의 위치를 찾고 해당 위치부터 upper보다 작은 값들을 모두 vector에 넣어 return하였습니다.

print에서는 root와 그 자식들의 key값을 출력하는 연산을 구현하였습니다. 파일이름을 인수로 받아 ostream을 하나 만들어주고 root의 key값들을 output이라는 함수를 이용해 해당 ostream으로 출력해주었습니다. 그리고 그 children들의 key값들을 같은 방식으로 출력해주었습니다. 해당 함수에서는 root가 leaf일 때, 그리고 children이 leaf일 때, non-leaf일때를 모두 고려하여 각각 다른 처리를 하도록 구현하였습니다.

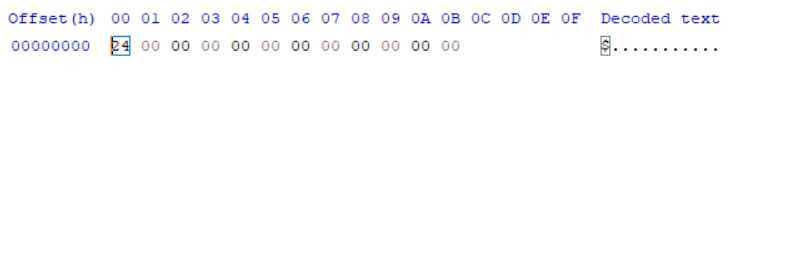
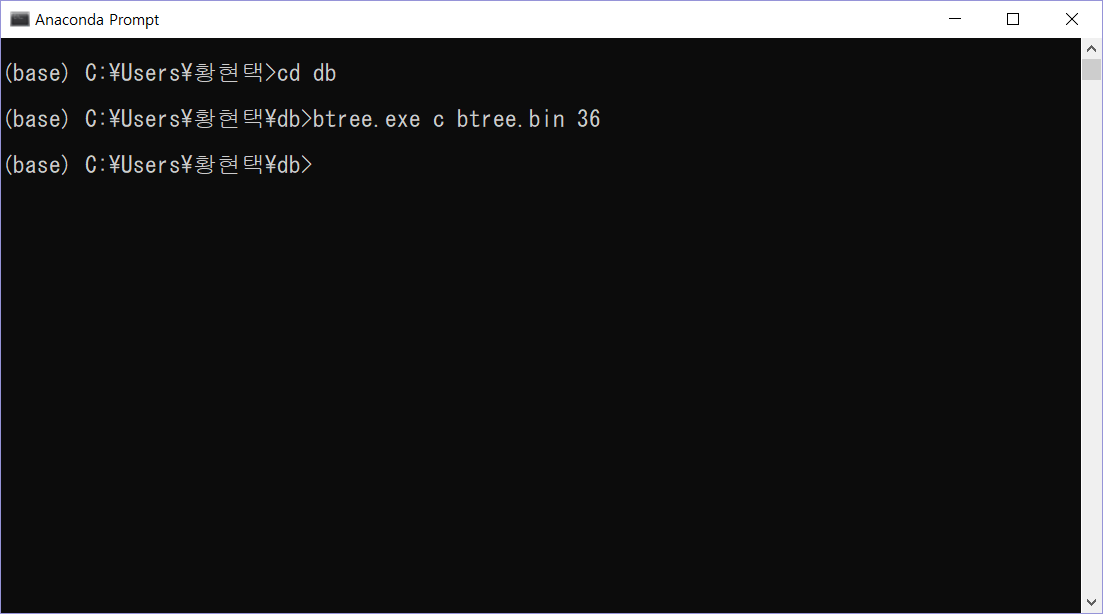
main 함수에서는 argv를 통해 명령어를 입력 받아 c,I,s,r,p를 각각의 상황에 맞게 구현하였습니다. c에서는 bp라는 포인터에 BPTree를 동적할당해 생성자를 호출하여 파일을 생성하는 작업을 구현하였습니다. i에서는 파일이 입력 파일이 끝날 때까지 한줄씩 입력 받아 ‘,’로 스트링을 parsing하여 각각 key와 id로 하여 insert함수를 호출하였습니다. 그리고 s에서는 key를 입력받아 find함수를 호출해 출력 파일에 기록하였으며, r에서는 한줄씩 입력받아 파일을 ‘,’로 parsing하여 lower와 upper에 넣어 range Search 함수를 호출한 뒤 파일에 기록하였습니다. 그리고 p에서는 print함수에 파일이름을 인수로 주어 해당 파일에 결과를 기록하도록 하였습니다.

1. 컴파일 환경 및 언어

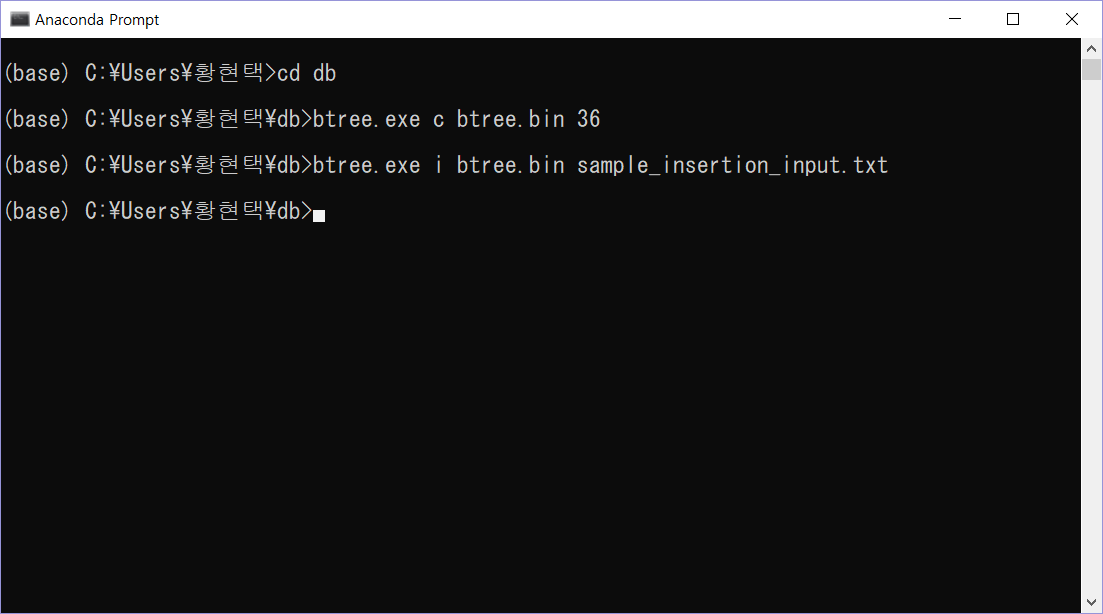
c++로 작성하였으며, windows10에서 visual studio 2019를 통해 컴파일 하였습니다.

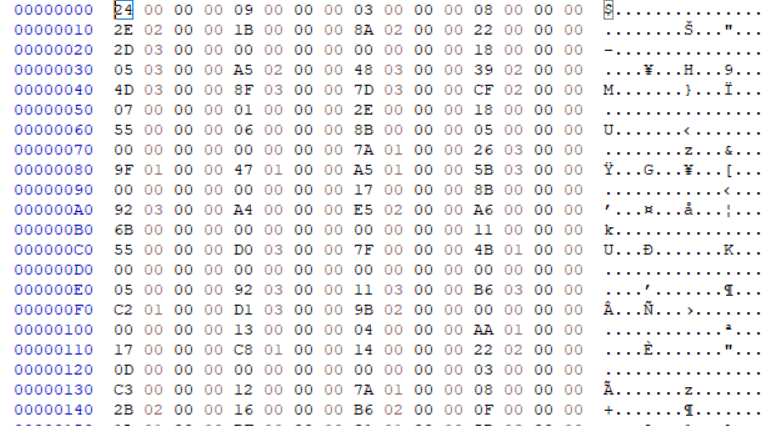
1. 출력 결과

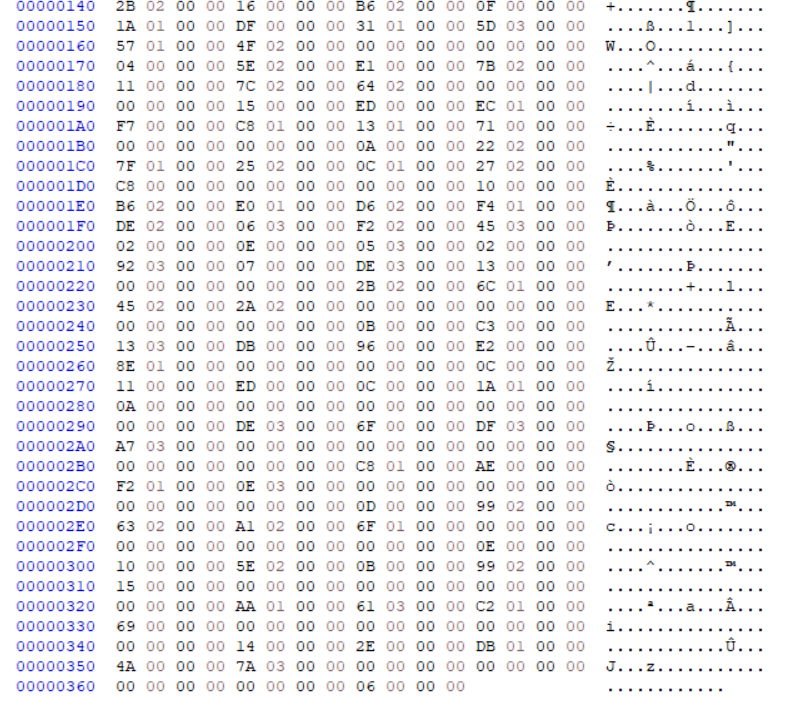
1. create



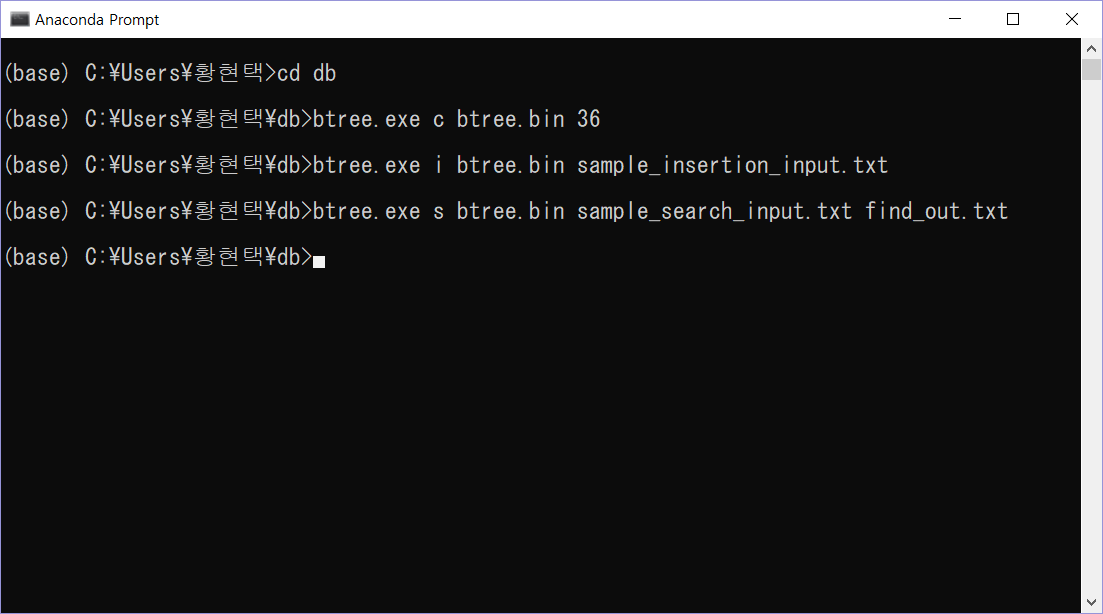
2. insert

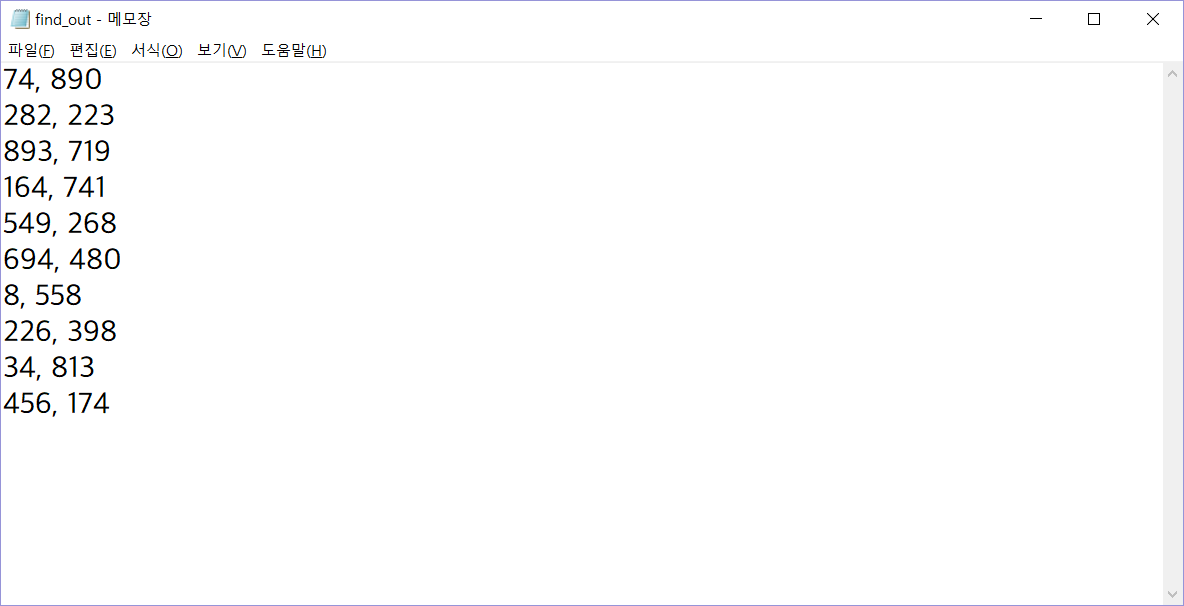




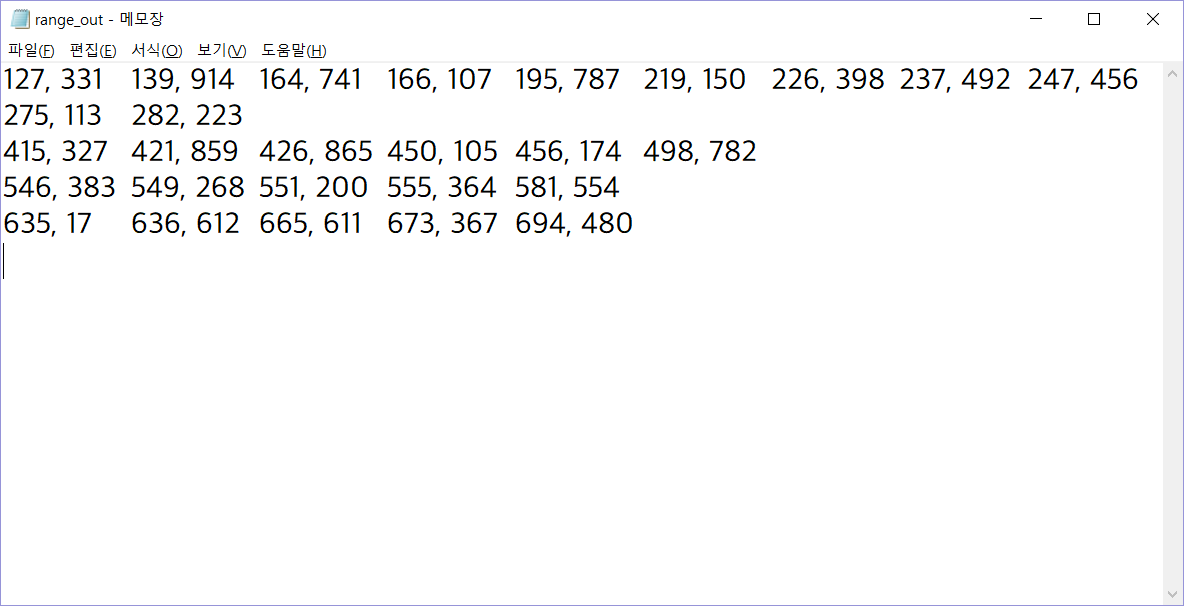
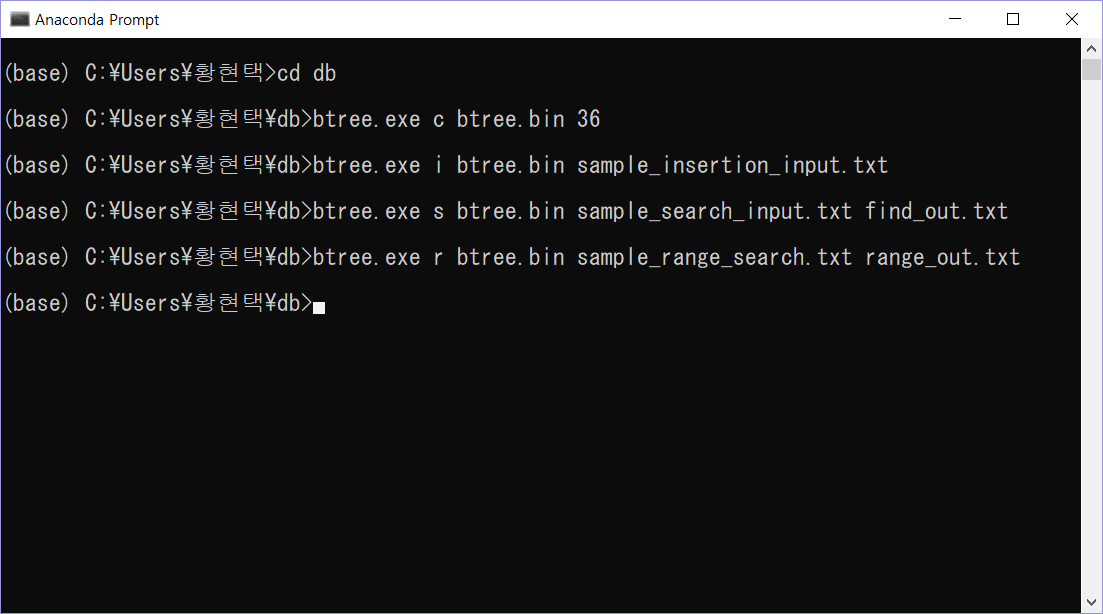


3. find





4. range search



5. print

