1. 구현한 것과 구현하지 않은 것

B+-tree만들기, 삽입, 검색, 범위검색, print모두 구현

1. 구현에 대한 간단한 설명

b+-tree를 만들 때 입력받은 사이즈, root번호, leaf노드를 제외한 트리의 높이, 블록에 들어갈 수 있는 int개수, 블록의 개수를 앞에 입력해준다.

삽입할 때 이진파일을 읽어오고 key, value를 받아와 b+-tree형태로 삽입해준다. split연산의 경우 btree class안에 포함되어 있고 나머지 연산은 모두 main함수 안에 있다. split연산은 노드의 크기가 입력받은 사이즈보다 커졌을 때 노드를 둘로 나누고 그 부모노드에 새로운 data를 삽입해주는 역할을 한다. split연산은 나누어지는 노드가 leaf노드인 경우와 non-leaf노드인 경우로 나누어져 있다. 다만 삽입할 때에는 매번 블록을 읽어오는 것이 아닌 모든 블록을 읽어온 후 명령을 수행한다.

필요한 블록은 블록번호, 파일 이름, 블록의 사이즈를 통해 배열형태로 미리 읽어온다. 블록을 읽고 쓰는 함수는 read\_block, write\_block이다. 이 함수는 검색, 범위검색, print연산에서 사용된다.

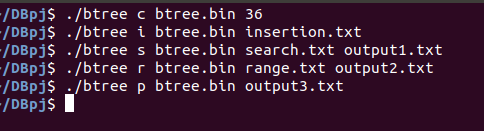
검색할 때 트리의 높이의 개수만큼 non\_leaf노드에서 탐색한 후 leaf 노드를 찾는다. 그다음 입력받은 key를 통해 key와 value를 찾아서 출력한다.

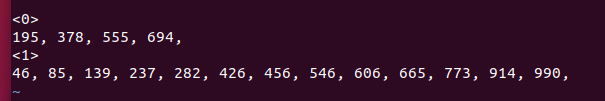
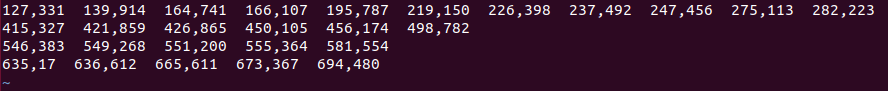
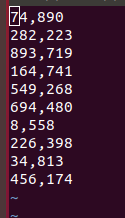
범위 검색할 때 트리의 높이의 개수만큼 non\_leaf노드에서 탐색한 후 leaf 노드를 찾는다. 그다음 입력받은 key를 통해 key와 value를 찾아서 출력한다. 범위에 있는 수가 아닐때까지 계속출력한다.

print연산은 높이가 0, 1, 2이상의 경우를 나누어 출력해준다. 높이가 0인경우 root노드가 leaf노드인 경우, 높이가 1인경우 root만 non-leaf노드이고 밑은 leaf노드인 경우이다. 높이가 2이상인 경우 모두 non-leaf인 경우이다.

1. 컴파일 및 실행 방법

btree.cpp파일을 실행하면서 동시에 명령어를 입력해주면 된다.





위에서부터 output1.txt, output2.txt, output3.txt 의 내용이다. insert, search, range는 모두 과제에 올라온 테스트 케이스 input내용이다.