과제 5: 인덱스 생성 및 이진 검색

1. 개요

지난 과제 4를 통해 생성한 레코드 파일을 이용하여 프로그램을 작성한다. 아래의 조건을 반드시 준수하여야 한다.

- 파일 I/O 연산은 system call 또는 C 라이브러리만을 사용한다.
- 제공되는 person.h와 person.c를 이용하여 아래의 (2) (3)의 기능을 완성한다.

(1) 'Person' 레코드 파일의 구조
- 지난 과제 4의 레코드 파일 구조와 모두 동일하다.
- 레코드 파일에 삭제 레코드가 존재할 수 있다.

(2) 인덱스 파일 생성 – person.c에서 createIndex() 구현
- 주어진 레코드 파일에서 각 레코드 (삭제 레코드 제외)를 읽어 '주민번호' 킷값과 레코드 주소 (페이지 번호 + 페이지 내 레코드 번호)를 추출하여 simple index file을 생성한다.
- 인덱스는 fixed-length record로 구성이 되고, 각 레코드는 세 개의 필드를 가진다. 첫 번째 필드는 '주민번호'로서 13 바이트를, 두 번째 필드는 '페이지 번호"로서 4 바이트를, 세 번째 필드는 '레코드 번호'로서 4 바이트를 갖는다. 따라서 인덱스 파일에서 레코드의 크기는 21 바이트이다.
- 인덱스 파일은 'Person' 레코드 파일과는 달리 페이지 단위가 아닌 레코드 단위로 저장되고 관리된다. 인덱스 파일은 크게 헤더 영역과 데이터 영역으로 나뉘며, 헤더 영역은 인덱스 파일에 저장되어 있는 레코드 개수를, 데이터 영역은 실제 레코드들을 저장한다. 여기서 헤더 영역의 크기는 4 바이트이다.
- 따라서, 과제 4에서 사용하였던 페이지 단위의 읽기(readPage())와 쓰기(writePage())를 굳이 새로운 기능을 위해 사용할 필요는 없다. 물론, 이 두 함수를 사용해도 무방하다. 각자 원하는 방식대로 인덱스 파일을 관리할 수 있다. 예를 들면, 인덱스 파일에 대해 레코드를 읽거나 쓸 때 페이지 번호와 레코드 번호를 인자로 갖는 레코드 단위의 읽기 또는 쓰기 함수를 만들어 사용할 수 있다.
- 사용자로부터 레코드 파일 이름과 인덱스 파일 이름을 입력 받고, 주어진 레코드 파일에서 레코드를 읽어서 인덱스 파일을 생성한다.
- 인덱스 파일은 주민번호에 대해 오름차순으로 정렬되어야 하며, 당연히 주민번호에 대해 중복은 허용하지 않는다.
- 오름차순 정렬을 구현하는 방법은 여러 가지가 있지만, 이번 과제에서는 그 효율성을 따지지 않는다. 즉, 오름차순 정렬 조건만 만족시키면 된다.

a.out i <record file name> <indexfile name>

<예제>
$ a.out i records.dat records.idx

옵션으로 i를 사용하며 실행 파일과 같은 디렉토리에 있는 'records.dat' 레코드 파일을 주면 ‘records.idx’ 라는 인덱스 파일을 생성한다. 명령의 수행 후 출력은 없다.

** 주의 1: 인덱스 파일에서 헤더 영역과 레코드의 모든 정수는 이진 정수(binary integer)로 저장한다 (레코드의 주민번호는 제외).
** 주위 2: 정상적인 레코드 파일이 주어진다고 가정하고 인덱스 파일을 생성한다.

(3) 이진 검색 (binary search) - person.c에서 binarysearch() 구현
- 일반적으로 simple index는 메인 메모리에 상주시켜 놓고 이진 검색을 수행하지만, 이번 과제에서는 메인 메모리 상주가 아니라 인덱스 파일에서 직접 필요한 레코드를 읽어서 이진 검색을 수행한다.
- 이진 검색을 할 때 키는 ‘주민번호’를 사용하며, 주어진 킷값을 만족하는 레코드의 주소를 인덱스 파일에서 찾고, 그 주소를 이용하여 레코드 파일에서 해당 레코드를 읽어서 화면에 출력한다.
- 인덱스 파일에서 킷값 검색을 할 때 반드시 이진 검색을 구현하여 사용하여야 한다.
- 이진 검색에 대한 알고리즘은 강의자료에 나와 있으므로 필요 시 참조할 수 있다.

a.out b <record file name> <index file name> <key value>

<예제>
$ a.out b records.dat records.idx “010223291234“
#reads:18
id=010223291234
name=Gildong Hong
age=20
addr=Seoul, Korea
phone=010-123-4567
email=gdhong@ssu.ac.kr

옵션으로 b를 사용하며, 킷값으로 주민번호 010223291234를 주면 해당 레코드의 주소 즉, 레코드 파일에서의 페이지 번호와 그 페이지 내의 레코드 번호를 ‘records.idx’ 인덱스 파일 상에서 이진 검색을 이용하여 검색한 후, 레코드 주소를 이용하여 ‘records.dat’ 레코드 파일에서 해당 레코드를 읽어서 화면에 출력한다. 출력 시에 이진 검색에서 수행한 비교 횟수(즉, 인덱스 파일에서의 레코드 read 수)도 출력한다 (즉, 위의 예제에서 ‘#reads:18’).

만약, 주어진 킷값을 만족하는 레코드가 존재하지 않는 경우 아래와 같이 출력한다.
$ a.out b records.dat records.idx “010223291234“
#reads:18
no persons

2. 개발 환경

- OS: Linux 우분투 버전 18.0.4
- 컴파일러: gcc 7.5

** 반드시 이 환경을 준수해야 하며, 이를 따르지 않아서 발생하는 불이익은 본인이 책임져야 함

3. 제출물

- 프로그래밍한 소스파일 person.c를 하위폴더 없이(최상위 위치에) zip파일로 압축하여 myclass.ssu.ac.kr 과제 게시판에 제출한다 (모든 제출 파일들의 파일명은 반드시 소문자로 작성하며, 제공된 person.h는 제출할 필요가 없음).
- 압축한 파일은 반드시 학번_5.zip (예시 20061084_5.zip)과 같이 작성하며, 여기서 5는 다섯 번째 과제임을 의미함

** 채점 프로그램상 오류가 날 수 있으니 꼭 위 사항을 준수하기 바라며, 이를 따르지 않아서 발행하는 불이익은 본인이 책임져야 함