정보검색(SearchEngine) IR 실습 및 과제

- 학번 : B354025
- 이름 : 정근화

동작 원리 정리

STEP 1.

- step1.cpp

```
$ make step1
$ ./step1

1. ir.docnames 파일을 읽는다. - ir.docnames 파일에는 읽어야할 문서 파일들의 이름이 기록되어 있다.
2. getDocNamenDocpos 를 통해 ir.docnames 의 문서 이름 하나와 위치를 가져온다.
3. 가져온 문서 이름 하나를 읽는다.
4. printAllWordsWithDocpos 를 통해 해당 문서를 단어 단위로 끊고 가ㄱ 단어를 문서의 위치와 함께 출력한다.
```

- **Result**

```
mit            0
university            0
located            0
near            0
harvard            0
university            0
harvard            5
university            5
in            5
city            5
of            5
boston            5
mit            10
in            10
city            10
of            10
boston            10
nonamein ir.docnames does not exist.
baltimore            22
has            22
johns            22
hopkins            22
university            22
find            27
bwi            27
```

```
airport           27
near           27
baltimore           27
and           27
johns           27
hopkins           27
university           27
int           32
main           32
int           32
i           32
j           32
cout           32
i           32
j           32
int           41
main           41
int           41
argc           41
char           41
argv           41
string           41
s           41
int           41
i           41
cout           41
s           41
i           41
홍익대학교는           50
마포구에           50
있다           50
mit           59
in           59
boston           59
홍익대학교는           59
마포구에           59
있다           59
```

STEP 2.

- sort

```
$ ./step1 | sort

1. STEP 1. 의 출력 결과를 Pipe redirection 을 통해 sort 한다.
2. 단어를 기준으로 오름차순 정렬된 결과가 출력된다.
```

- **Result**

```
nonamein ir.docnames does not exist.
airport           27
and           27
argc           41
argv           41
baltimore           22
baltimore           27
boston           5
boston           10
boston           59
bwi           27
char           41
city           5
city           10
cout           32
cout           41
find           27
harvard           0
harvard           5
has           22
hopkins           22
hopkins           27
i           32
i           32
i           41
i           41
in           5
in           10
in           59
int           32
int           32
int           41
int           41
int           41
j           32
j           32
johns           22
johns           27
located           0
main           32
main           41
mit           0
mit           10
mit           59
near           0
near           27
of           5
of           10
s           41
s           41
string           41
university           0
university           0
```

```
university            5
university           22
university           27
마포구에           50
마포구에           59
있다           50
있다           59
홍익대학교는           50
홍익대학교는           59
```

STEP 3.

- step3.cpp

```
$ make step3
$ ./step1 | sort | ./step3

1. step3 은 ./step1 | sort 의 결과가 stdout 으로 나온다는 가정 하에
   Pipe redirection 을 사용하여 cin(stdin) 입력을 받는 것으로 시작한다.
2. step2 의 결과를 가ㄱ Line 별로 단어와 문서 위치를 읽는다.
3-1. 정렬이 되어 있으므로 이전 라인과 단어와 문서 위치가 가ㅌ다면 해당 포스팅에서 해당 단어의 빈도 만++ 해준다.
3-2. 만약 이전 라인과 단어는 가ㅌ지만 문서 위치가 다르다면 이전까지의 문서(Posting)를 라인에 추가하고 새 포스팅을 초기화한다.
3-3. 만약 단어가 바뀌었다면 해당 단어의 line의 문서(Posting) 을 추가한 후 출력한다. 또한 새 단어와 포스팅을 초기화한다.
4. 3.의 과정을 모두 마치면 (단어, 관련 문서 수, 총 빈도, 가ㄱ 문서의 위치와 문서내 빈도) 정보가 출력된다.
```

- **Result**

```
nonamein ir.docnames does not exist.
airport 1 1  27 1
and 1 1  27 1
argc 1 1  41 1
argv 1 1  41 1
baltimore 2 2  22 1 27 1
boston 3 3  5 1 10 1 59 1
bwi 1 1  27 1
char 1 1  41 1
city 2 2  5 1 10 1
cout 2 2  32 1 41 1
find 1 1  27 1
harvard 2 2  0 1 5 1
has 1 1  22 1
hopkins 2 2  22 1 27 1
i 2 4  32 2 41 2
in 3 3  5 1 10 1 59 1
int 2 5  32 2 41 3
```

```
j 1 2  32 2
johns 2 2  22 1 27 1
located 1 1  0 1
main 2 2  32 1 41 1
mit 3 3  0 1 10 1 59 1
near 2 2  0 1 27 1
of 2 2  5 1 10 1
s 1 2  41 2
string 1 1  41 1
university 4 5  0 2 5 1 22 1 27 1
마포구에 2 2  50 1 59 1
있다 2 2  50 1 59 1
홍익대학교는 2 2  50 1 59 1
```

STEP 4.

- step4.cpp

```
$ make step4
$ ./step1 | sort | ./step3 | ./step4

0. ir.words, ir.postings, ir.info, ir.dictionary 파일 생성합니다.
1. 기존의 파일들이 존재한다면 trunc 하여 지웁니다.
   단 ir.info 파일은 step1 에서 document 의 개수가 이미 저장이 되어 있으므로 지우지 않습니다.
2. 이전 step3 의 결과를 바탕으로 한 단어 씩 읽습니다.
3. 해당 단어를 ir.words 에 기록하고 lineCnt와 inLineCnt를 세어 전체 행과 irwords 의 열의 개수를 카운트합니다.
4. 다음 두 숫자를 읽습니다 (해당 단어의 문서 개수와 총 빈도)
5. ir.postings 를 열고 Dict_Term의 poststart(ir.postings 의 첫 위치) 와 numposts, idf 를 저장합니다.
   이때 maxIdf 도 계속 확인해주어 계산합니다.
6. 4에서 읽은 문서개수를 바탕으로 while을 돌며 가ㄱ가ㄱ 문서위치와 빈도를 vector<Posting>에 저장합니다.
7. 해당 vector<Posting>을 이진 형태로 ir.postings 에 기록합니다.
8. vector<Dict_term> 에 2-7 을 거쳐 생성한 Dict_term 을 저장합니다.
9. 2 - 8 까지의 과정을 반복하여 step3의 결과를 모두 처리합니다.
10. 위 과정을 반복하며 축적된 전체 단어 개수와 maxIdf 가ㅅ을 ir.info에 기록합니다.
11. 위 과정을 반복하며 축적된 vector<Dict_term>을 이진형태로 ir.dictionary 에 기록합니다.
```

- **Result**

```
# ir.words

airport and argc argv baltimore
boston bwi char city cout
find harvard has hopkins i
in int j johns located
```

```
main mit near of s
string university 마포구에 있다 홍익대학교는


# ir.postings --> hexdump 로 보기

  Offset: 00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F
00000000: 1B 00 00 00 01 00 00 00 1B 00 00 00 01 00 00 00
................
00000010: 29 00 00 00 01 00 00 00 29 00 00 00 01 00 00 00
).......).......
00000020: 16 00 00 00 01 00 00 00 1B 00 00 00 01 00 00 00
................
00000030: 05 00 00 00 01 00 00 00 0A 00 00 00 01 00 00 00
................
00000040: 3B 00 00 00 01 00 00 00 1B 00 00 00 01 00 00 00
;...............
00000050: 29 00 00 00 01 00 00 00 05 00 00 00 01 00 00 00
).............
00000060: 0A 00 00 00 01 00 00 00 20 00 00 00 01 00 00 00
................
00000070: 29 00 00 00 01 00 00 00 1B 00 00 00 01 00 00 00
).............
00000080: 00 00 00 00 01 00 00 00 05 00 00 00 01 00 00 00
................
00000090: 16 00 00 00 01 00 00 00 16 00 00 00 01 00 00 00
................
000000a0: 1B 00 00 00 01 00 00 00 20 00 00 00 02 00 00 00
................
000000b0: 29 00 00 00 02 00 00 00 05 00 00 00 01 00 00 00
).............
000000c0: 0A 00 00 00 01 00 00 00 3B 00 00 00 01 00 00 00
........;.......
000000d0: 20 00 00 00 02 00 00 00 29 00 00 00 03 00 00 00
........).......
000000e0: 20 00 00 00 02 00 00 00 16 00 00 00 01 00 00 00
................
000000f0: 1B 00 00 00 01 00 00 00 00 00 00 00 01 00 00 00
................
00000100: 20 00 00 00 01 00 00 00 29 00 00 00 01 00 00 00
........).......
00000110: 00 00 00 00 01 00 00 00 0A 00 00 00 01 00 00 00
................
00000120: 3B 00 00 00 01 00 00 00 00 00 00 00 01 00 00 00
;...............
00000130: 1B 00 00 00 01 00 00 00 05 00 00 00 01 00 00 00
................
00000140: 0A 00 00 00 01 00 00 00 29 00 00 00 02 00 00 00
........).......
00000150: 29 00 00 00 01 00 00 00 00 00 00 00 02 00 00 00
).............
00000160: 05 00 00 00 01 00 00 00 16 00 00 00 01 00 00 00
................
00000170: 1B 00 00 00 01 00 00 00 32 00 00 00 01 00 00 00
```

```
........2.......
00000180: 3B 00 00 00 01 00 00 00 32 00 00 00 01 00 00 00
;.......2.......
00000190: 3B 00 00 00 01 00 00 00 32 00 00 00 01 00 00 00
;.......2.......
000001a0: 3B 00 00 00 01 00 00 00                         ;.......


# ir.info

9 30 3.16993


# ir.dictionary --> hexdump 로 보기

  Offset: 00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F
00000000: 00 00 00 00 00 00 00 00 01 00 00 00 96 7F 00 00
................
00000010: 68 BD 9F A3 01 5C 09 40 08 00 00 00 08 00 00 00
h=.#.\.@........
00000020: 01 00 00 00 96 7F 00 00 68 BD 9F A3 01 5C 09 40
........h=.#.\.@
00000030: 0C 00 00 00 10 00 00 00 01 00 00 00 96 7F 00 00
................
00000040: 68 BD 9F A3 01 5C 09 40 11 00 00 00 18 00 00 00
h=.#.\.@........
00000050: 01 00 00 00 96 7F 00 00 68 BD 9F A3 01 5C 09 40
........h=.#.\.@
00000060: 16 00 00 00 20 00 00 00 02 00 00 00 96 7F 00 00
................
00000070: 68 BD 9F A3 01 5C 01 40 21 00 00 00 30 00 00 00
h=.#.\.@!...0...
00000080: 03 00 00 00 96 7F 00 00 68 BD 9F A3 01 5C F9 3F
........h=.#.\y?
00000090: 28 00 00 00 48 00 00 00 01 00 00 00 96 7F 00 00
(...H...........
000000a0: 68 BD 9F A3 01 5C 09 40 2C 00 00 00 50 00 00 00
h=.#.\.@,...P...
000000b0: 01 00 00 00 96 7F 00 00 68 BD 9F A3 01 5C 09 40
........h=.#.\.@
000000c0: 31 00 00 00 58 00 00 00 02 00 00 00 96 7F 00 00
1...X...........
000000d0: 68 BD 9F A3 01 5C 01 40 36 00 00 00 68 00 00 00
h=.#.\.@6...h...
000000e0: 02 00 00 00 96 7F 00 00 68 BD 9F A3 01 5C 01 40
........h=.#.\.@
000000f0: 3C 00 00 00 78 00 00 00 01 00 00 00 96 7F 00 00
<...x...........
00000100: 68 BD 9F A3 01 5C 09 40 41 00 00 00 80 00 00 00
h=.#.\.@A.......
00000110: 02 00 00 00 96 7F 00 00 68 BD 9F A3 01 5C 01 40
........h=.#.\.@
00000120: 49 00 00 00 90 00 00 00 01 00 00 00 96 7F 00 00
I...............
```

```
00000130: 68 BD 9F A3 01 5C 09 40 4D 00 00 00 98 00 00 00
h=.#.\.@M.......
00000140: 02 00 00 00 96 7F 00 00 68 BD 9F A3 01 5C 01 40
........h=.#.\.@
00000150: 55 00 00 00 A8 00 00 00 02 00 00 00 96 7F 00 00    U...
(...........
00000160: 68 BD 9F A3 01 5C 01 40 58 00 00 00 B8 00 00 00
h=.#.\.@X...8...
00000170: 03 00 00 00 96 7F 00 00 68 BD 9F A3 01 5C F9 3F
........h=.#.\y?
00000180: 5B 00 00 00 D0 00 00 00 02 00 00 00 96 7F 00 00
[...P...........
00000190: 68 BD 9F A3 01 5C 01 40 5F 00 00 00 E0 00 00 00
h=.#.\.@_...`...
000001a0: 01 00 00 00 96 7F 00 00 68 BD 9F A3 01 5C 09 40
........h=.#.\.@
000001b0: 61 00 00 00 E8 00 00 00 02 00 00 00 96 7F 00 00
a...h...........
000001c0: 68 BD 9F A3 01 5C 01 40 67 00 00 00 F8 00 00 00
h=.#.\.@g...x...
000001d0: 01 00 00 00 96 7F 00 00 68 BD 9F A3 01 5C 09 40
........h=.#.\.@
000001e0: 70 00 00 00 00 01 00 00 02 00 00 00 96 7F 00 00
p...............
000001f0: 68 BD 9F A3 01 5C 01 40 75 00 00 00 10 01 00 00
h=.#.\.@u.......
00000200: 03 00 00 00 96 7F 00 00 68 BD 9F A3 01 5C F9 3F
........h=.#.\y?
00000210: 79 00 00 00 28 01 00 00 02 00 00 00 96 7F 00 00    y...
(...........
00000220: 68 BD 9F A3 01 5C 01 40 7E 00 00 00 38 01 00 00
h=.#.\.@~...8...
00000230: 02 00 00 00 96 7F 00 00 68 BD 9F A3 01 5C 01 40
........h=.#.\.@
00000240: 81 00 00 00 48 01 00 00 01 00 00 00 96 7F 00 00
....H...........
00000250: 68 BD 9F A3 01 5C 09 40 84 00 00 00 50 01 00 00
h=.#.\.@....P...
00000260: 01 00 00 00 96 7F 00 00 68 BD 9F A3 01 5C 09 40
........h=.#.\.@
00000270: 8B 00 00 00 58 01 00 00 04 00 00 00 96 7F 00 00
....X...........
00000280: D1 7A 3F 47 03 B8 F2 3F 96 00 00 00 78 01 00 00    Qz?
G.8r?....x...
00000290: 02 00 00 00 96 7F 00 00 68 BD 9F A3 01 5C 01 40
........h=.#.\.@
000002a0: A3 00 00 00 88 01 00 00 02 00 00 00 96 7F 00 00
#...............
000002b0: 68 BD 9F A3 01 5C 01 40 AA 00 00 00 98 01 00 00
h=.#.\.@*.......
000002c0: 02 00 00 00 96 7F 00 00 68 BD 9F A3 01 5C 01 40
........h=.#.\.@
```

Printdict

- printdict.cpp

```
$ make printdict
$ ./printdict

Step4 에서 생성한 4개의 files (ir.words, ir.postings, ir.info, ir.dictionary)
과 ir.docnames
를 사용해서 출력합니다.

1. ir.dictionary 를 읽어 가ㄱ Dict_Terms 구조체의 정보를 바탕으로 하위 ir.postings
와 ir.words 를 읽습니다.
2. ir.postings 가 가진 docpos 를 사용해 ir.docnames 를 읽습니다.
3. <collection> 은 ir.info 를 따록 읽어서 출력합니다.
```

- **Result**

```
airport appeared 1 time(s) in 1 document(s) [ idf = 3.16993 ]
   doc5 1
and appeared 1 time(s) in 1 document(s) [ idf = 3.16993 ]
   doc5 1
argc appeared 1 time(s) in 1 document(s) [ idf = 3.16993 ]
   doc7.cpp 1
argv appeared 1 time(s) in 1 document(s) [ idf = 3.16993 ]
   doc7.cpp 1
baltimore appeared 2 time(s) in 2 document(s) [ idf = 2.16993 ]
   doc4 1   doc5 1
boston appeared 3 time(s) in 3 document(s) [ idf = 1.58496 ]
   doc2 1   doc3 1   doc9.kor 1
bwi appeared 1 time(s) in 1 document(s) [ idf = 3.16993 ]
   doc5 1
char appeared 1 time(s) in 1 document(s) [ idf = 3.16993 ]
   doc7.cpp 1
city appeared 2 time(s) in 2 document(s) [ idf = 2.16993 ]
   doc2 1   doc3 1
cout appeared 2 time(s) in 2 document(s) [ idf = 2.16993 ]
   doc6.cpp 1   doc7.cpp 1
find appeared 1 time(s) in 1 document(s) [ idf = 3.16993 ]
   doc5 1
harvard appeared 2 time(s) in 2 document(s) [ idf = 2.16993 ]
   doc1 1   doc2 1
has appeared 1 time(s) in 1 document(s) [ idf = 3.16993 ]
   doc4 1
hopkins appeared 2 time(s) in 2 document(s) [ idf = 2.16993 ]
   doc4 1   doc5 1
i appeared 4 time(s) in 2 document(s) [ idf = 2.16993 ]
   doc6.cpp 2   doc7.cpp 2
in appeared 3 time(s) in 3 document(s) [ idf = 1.58496 ]
   doc2 1   doc3 1   doc9.kor 1
```

```
int appeared 5 time(s) in 2 document(s) [ idf = 2.16993 ]
   doc6.cpp 2   doc7.cpp 3
j appeared 2 time(s) in 1 document(s) [ idf = 3.16993 ]
   doc6.cpp 2
johns appeared 2 time(s) in 2 document(s) [ idf = 2.16993 ]
   doc4 1   doc5 1
located appeared 1 time(s) in 1 document(s) [ idf = 3.16993 ]
   doc1 1
main appeared 2 time(s) in 2 document(s) [ idf = 2.16993 ]
   doc6.cpp 1   doc7.cpp 1
mit appeared 3 time(s) in 3 document(s) [ idf = 1.58496 ]
   doc1 1   doc3 1   doc9.kor 1
near appeared 2 time(s) in 2 document(s) [ idf = 2.16993 ]
   doc1 1   doc5 1
of appeared 2 time(s) in 2 document(s) [ idf = 2.16993 ]
   doc2 1   doc3 1
s appeared 2 time(s) in 1 document(s) [ idf = 3.16993 ]
   doc7.cpp 2
string appeared 1 time(s) in 1 document(s) [ idf = 3.16993 ]
   doc7.cpp 1
university appeared 5 time(s) in 4 document(s) [ idf = 1.16993 ]
   doc1 2   doc2 1   doc4 1   doc5 1
마포구에 appeared 2 time(s) in 2 document(s) [ idf = 2.16993 ]
   doc8.kor 1   doc9.kor 1
있다 appeared 2 time(s) in 2 document(s) [ idf = 2.16993 ]
   doc8.kor 1   doc9.kor 1
홍익대학교는 appeared 2 time(s) in 2 document(s) [ idf = 2.16993 ]
   doc8.kor 1   doc9.kor 1

<Collection Summary>
#Docs = 9  #Words = 30  #Max.IDF = 3.16993
```

Consult

- consult.cpp 는 시가ㄴ 부족으로 작업하지 못했습니다.