SWE2024-41: System Programming Assignment (Spring 2023)

Programming Assignment #1

Due: April 23th, 11:59 PM

1. Introduction

과제를 통해 자료구조에 대한 이해와 File I/O 사용에 익숙해진다.

2. Specification

이 과제의 목표는, shell command로 입력된 영화 시나리오 파일을 읽고 특정 검색 기능을 수행하는 코드를 만들어 보는 것이다. 프로그램이 시작된 후, 프로그램은 사용자의 키워드 입력(최대 512 Bytes)을 기다리며 아래에 언급된 검색 기능을 수행한다.

① Searching single word locations

프로그램에서 하나의 단어를 입력 받는 경우, 단어의 위치를 찾는다.

- 영화 시나리오에서 입력 받은 단어가 포함된 행을 찾아 다음의 형식으로 `stdout`에 출력한다.

```
"[line number]:[start index of the word]"
```

② Searching several words locations

프로그램이 여러 단어(공백 하나로 구분)를 입력 받는 경우, 입력된 모든 단어를 포함하는 행을 찾는다.

- 영화 시나리오에서 입력 받은 단어가 모두 포함된 행을 찾아 `stdout`에 출력한다.

```
"[line number]"
```

③ **Searching several consecutive words locations**

프로그램에서 ""로 싸여진 구문을 입력 받는 경우, 구문의 위치를 찾는다.

● 구문 내에는 개행문자가 존재하지 않는다.

● 구문 내에는 연속한 공백 및 탭이 존재할 수 있다.

● 영화 시나리오에서 입력 받은 구문이 포함된 행을 찾아 다음의 형식으로 stdout에 출력한다.

"[line number]:[start index of the phrase]"

● 입력 파일의 하나의 문자열에 대하여 중복되는 구문들이 존재한다면, 모든 구문의 위치를 아래의 예시처럼 출력한다.

Example of Input File Line: HA HA HA HA /* first line of input file */

```
$ "HA HA"
1:0 1:3 1:6
```

④ **Searching simple regular expressing keyword locations**

프로그램에서 [단어1]*[단어2]로 구성된 "두 개의 단어"를 입력 받는 경우, [단어1]과 [단어2] 사이에 하나 이상의 문자가 포함된 행을 찾는다.

● [word1]과 *, 그리고 *과 [word2] 사이에는 빈 칸이 없다.

● [word2]*[word1]의 경우는 고려하지 않는다

● 영화 시나리오에서 위의 설명에 해당하는 행을 찾아 stdout에 출력한다.

"[line number]"

⑤ **Supplementary explanations**

● **line number**와 **start index of the word (or phrase)**에 대한 설명은 다음과 같다.

* **line number** – 단어(혹은 구문이나 키워드)를 포함하는 행의 번호이며, 값은 1부터 시작한다. **line number는 어떠한 값도 없는 빈 행 또한 포함한다.**

* **start index of the word (or phrase)** – 행에 있는 단어(혹은 구문)의 시작 번호이며, 값은 0

부터 시작해서 하나의 아스키 코드 표에 존재하는 1 Byte 문자 별로 1씩 증가한다. **빈 칸과 writing symbol 또한 포함한다.**

● 입력 받은 단어가 영화 시나리오에 여러 번 나타나면, 해당하는 행을 모두 출력해야 한다.

- ①, ③의 경우, 같은 행에 존재하더라도 **모든 위치를 출력해야 한다.**
- ②, ④의 경우, 중복된 행에 대해서 한 번만 출력하면 된다.

● **다수의 검색 결과가 존재하는 경우, 하나의 검색 결과 위치 뒤에 빈 칸이 추가되어야 한다. 마지막 검색 결과 끝에는 빈 칸이 아닌, 개행문자가 존재해야 한다. 아래의 예시는 이를 나타낸다.**

```
$ "he is"
1955:33 2315:42 5885:22
(New Line)
```

● 입력에서 " 또는 * 문자에 대해서는 다음과 같이 분류한다.

- 입력으로는 "와 *가 동시에 포함되지 않는다. 따라서, 해당 테스트는 진행하지 않는다.
- 입력에 " 문자가 포함되어 있을 경우, ③에 해당한다.
- 입력에 * 문자가 포함되어 있을 경우, ④에 해당한다.
- 입력이 " 와 * 문자가 모두 포함되어 있지 않은 2개 이상의 단어로 구성된 경우, ②에 해당한다.
- 입력이 " 와 * 문자가 모두 포함되어 있지 않은 1개의 단어로 구성된 경우, ①에 해당한다.

● 검색 결과는 입력 파일의 첫 행/열에 대한 결과부터 순차적으로 출력되어야 한다.

● 단어를 찾을 파일은 커맨드 라인 argument로 받는다. (argv[1])

● 입력 파일은 시스템의 메모리보다 크다고 가정하며, 입력 파일 내의 가장 긴 문자열의 크기는 시스템의 메모리보다 작다고 가정한다. 즉, 입력 파일을 한번에 전부 메모리 상에 위치시킬 수 없다.

● 입력 파일 및 키워드의 문자는 아스키 코드 범위 내의 문자만 포함한다.

- 입력 파일의 문자열은 연속해서 하나 이상의 빈 칸이 존재할 수 있다.
- 입력 파일은 빈 칸으로 시작하지 않는다.
- 키워드는 빈 칸으로 시작하거나 끝나지 않는다.
- **단어 입력에 대한 세부사항은 "6. Example" 참조**
- 코드는 입력 대기를 무한히 실행하며, 입력 키워드가 "PA1EXIT"인 경우 실행을 종료한다.

3. Score Policy

- **100점 만점을 기준으로 하며 위에서 설명한 형식을 따라야 하며, 그렇지 않으면 점수를 받을 수 없다.**

- **마감 이후 제출은 하루 단위로 10점씩 감점이 적용된다. (10일이 지난 후, 제출 시 0점)**

- 어느 정도의 discussion은 허용하지만, 소스코드는 스스로 작성해야 한다.

- 채점은 각 테스트케이스 별 통과 유무로 진행되며, 테스트케이스 통과 시 할당된 점수가 총점에 포함되며, 테스트케이스를 통과하지 못한 경우 할당된 점수는 총점에 포함되지 않는다. 테스트케이스 별 부분점수는 존재하지 않는다.

- 예시로 제공된 입력 파일을 포함하여 다수의 입력 파일이 채점에 사용되며, 각 입력 파일의 용량의 범위는 수십 GB 까지 존재할 수 있다.

- 채점 서버에서 테스트케이스 별 종료 시간이 5분 이상이 소요되는 경우, 해당 테스트케이스에 대한 점수는 제공되지 않는다. 채점은 채점 서버를 기준으로, 종료까지 10초 이내의 테스트케이스만 사용한다.

- 채점 서버의 메모리는 1 GB로 가정한다.

4. Restriction

- 이 과제는 **리눅스 환경**에서 구현하는 것을 기준으로 한다.

- 사용 가능한 헤더 파일은 아래의 파일들로 제한된다. 다른 헤더를 사용 시 0점 처리된다.

  (1) 최소 한 개의 **직접 구현한** 헤더 파일 (필수)

  (2) `unistd.h fcntl.h stdlib.h sys/types.h sys/stat.h errno.h`

- `Makefile`을 통해 생성되는 바이너리 실행파일의 이름은 `pa1`으로 (소문자) 설정한다.

- 단어: 빈 칸으로 구분되어지는 문자열이며, 대소문자를 구분하지 않는다.

- 단어 검색에 사용될 수 있는 단어의 예시는 다음과 같다.

  e.g.) god, and, adam, brother's, priests', kirjath-arba, sons', score:1

- 구문: 개행으로 구분되어지는 문자열이며, 대소문자를 구분하지 않는다.

- 빈 칸: 탭, 공백, 개행을 의미한다.

- Writing symbol을 포함하는 단어들은 서로 다른 단어들로 간주된다. **예를 들어, he를 검색하면 her, he's가 아닌 오직 he만 검색 결과에 포함해야 한다.**

5. Submit instructions

pa1 (directory)

- ***.c (C code files)**
- ***.h (Header files)**
- **Makefile**

Submit using the command:

```
$ ~swe2024-41_23s/bin/submit pa1 pa1
```

6. Example (빨간색으로 강조 표시된 단어가 입력으로 주어짐)

```
$ ./pa1 500-Days-of-Summer_s.txt

500 days

6 6419

he is

50 1039 1822 1955 2256 2315 3494 3503 4353 4360 4445 4831 5101 5885
6325

"he is"

1955:33 2315:42 5885:22

he*is

2315 4445 5101

she he

1370 1512 1513 3423 3473 3478 3550 4255 4510 4515 5413 5672 6154
6188

loved

106:30 1122:9 1150:24 3961:25 4739:17 5921:28

…
```

```
$ ./pa1 500-Days-of-Summer_s.txt > result.out

he

hey

she

tom

summer is

"summer is"

landscape

together

we*her

no much

immediately

no*much

quarterback

we

PA1EXIT


$ diff -bsq result.out answer.out
Files result.out and answer.out are identical
$ diff result.out answer.out
$
```