

과제 구현 보고서

학 과
학 번
이 름
제출일

컴퓨터공학과
201211704
김기홍
2015.10.

개요

기본 입력은 the_gold_bug.txt로 하였으며 이 보고서는 프로그램의 실행화면 및 구현과정의 대략적인 설명 그리고 코드 파일(첨부)로 이루어져 있다.

1. a-z까지의 문자 개수를 세어 문자사용 비율을 계산하는 프로그램을 작성하라.
결과는 text.res1에 저장한다.

1) 저수준의 입출력을 사용

구현 과정 : open(), close(), read(), write()함수만을 사용하여 원하는 결과를 얻었다. 알파벳(대,소문자 구별x)에 해당하는 크기가 26인 정수형 배열을 만들고 a는 인덱스 0, b는 인덱스 1 이런 방식으로 매핑을 해주었다. 각 배열의 인덱스 값은 알파벳의 사용빈도를 카운팅 하였다.

사용 비율을 계산하기 위해 각각의 인덱스 값에 배열의 모든 값을 더한 sum 변수를 나누어 비율을 구하였고 그 비율을 gcvt함수(실수를 문자열로 변환)를 사용하여 res1 파일에 write할 수 있었다.

<pre>hong@hong ~/Desktop/SYSProg/report/low_level \$ a : 7.715% b : 1.771% c : 2.617% d : 4.331% e : 13.12% f : 2.386% g : 1.968% h : 5.801% i : 7.189% j : 0.192% k : 0.608% l : 4.019% m : 2.571% n : 6.729% o : 7.219% p : 1.948% q : 0.103% r : 5.600% s : 6.058% t : 9.447% u : 3.202% v : 0.906% w : 2.239% x : 0.202% y : 1.968% z : 0.074% hong@hong ~/Desktop/SYSProg/report/low_level \$</pre>	<pre>the_gold_bug.res1 x a : 7.715% b : 1.771% c : 2.617% d : 4.331% e : 13.12% f : 2.386% g : 1.968% h : 5.801% i : 7.189% j : 0.192% k : 0.608% l : 4.019% m : 2.571% n : 6.729% o : 7.219% p : 1.948% q : 0.103% r : 5.600% s : 6.058% t : 9.447% u : 3.202% v : 0.906% w : 2.239% x : 0.202% y : 1.968% z : 0.074%</pre>
--	--

2) 고수준의 입출력을 사용

구현 과정 : FILE *포인터와 fopen() fclose() fgetc() fprintf()함수로 구현하였으며 문자를 파일로부터 읽어 들이는 작업을 할 때에는 저수준의 함수와 방법적 차이는 없어서 차이점을 잘 알 수 없었지만 파일에 쓰는 작업에 대해서는 fprintf()함수를 통해 데이터 형을 정하고 틀을 잡을 수 있기 때문에 비교적 아주 손쉽게 구현할 수 있었다.

<pre> hong@hong ~/Desktop/SYSProg/report/high_level \$ a : 7.715% b : 1.771% c : 2.618% d : 4.331% e : 13.125% f : 2.386% g : 1.969% h : 5.801% i : 7.189% j : 0.192% k : 0.608% l : 4.020% m : 2.572% n : 6.729% o : 7.220% p : 1.948% q : 0.104% r : 5.600% s : 6.058% t : 9.448% u : 3.202% v : 0.906% w : 2.240% x : 0.203% y : 1.969% z : 0.075% hong@hong ~/Desktop/SYSProg/report/high_level \$ </pre>	<pre> the_gold_bug.res1 x a : 7.715% b : 1.771% c : 2.618% d : 4.331% e : 13.125% f : 2.386% g : 1.969% h : 5.801% i : 7.189% j : 0.192% k : 0.608% l : 4.020% m : 2.572% n : 6.729% o : 7.220% p : 1.948% q : 0.104% r : 5.600% s : 6.058% t : 9.448% u : 3.202% v : 0.906% w : 2.240% x : 0.203% y : 1.969% z : 0.075% </pre>
---	---

2) lex를 이용한 입출력

구현 과정 : 익숙하지 않아서 익히는 데 조금 시간은 걸렸지만 막상 구현하고 나니 lex의 편리함에 아주 감탄할 정도였다. yyin 과 yyout을 통해 파일을 열고 닫을 수 있었으며, 가장 인상적이었던 점은 읽는 함수가 따로 필요하지 않았다는 점이다. 파일을 연 상태에서 yylex()를 호출해주면 미리 기술된 lex에 의해 파일의 끝까지 읽어내 주었다. 그래서 내가 따로 할 일은 lex에서 알파벳 하나당 매핑된 배열 인덱스 값에 카운팅을 해주는 일과 그 배열의 값을 이용해 res1 파일에 써주는 일 밖에 없었다.

<pre> hong@hong ~/Desktop/SYSProg/report/lex_yacc \$ a : 7.715% b : 1.771% c : 2.618% d : 4.331% e : 13.125% f : 2.386% g : 1.969% h : 5.801% i : 7.189% j : 0.192% k : 0.608% l : 4.020% m : 2.572% n : 6.729% o : 7.220% p : 1.948% q : 0.104% r : 5.600% s : 6.058% t : 9.448% u : 3.202% v : 0.906% w : 2.240% x : 0.203% y : 1.969% z : 0.075% hong@hong ~/Desktop/SYSProg/report/lex_yacc \$ </pre>	<pre> the_gold_bug.res1 x a : 7.715% b : 1.771% c : 2.618% d : 4.331% e : 13.125% f : 2.386% g : 1.969% h : 5.801% i : 7.189% j : 0.192% k : 0.608% l : 4.020% m : 2.572% n : 6.729% o : 7.220% p : 1.948% q : 0.104% r : 5.600% s : 6.058% t : 9.448% u : 3.202% v : 0.906% w : 2.240% x : 0.203% y : 1.969% z : 0.075% </pre>
---	---

2. 사용된 단어의 종류와 사용 횟수를 출력하는 프로그램을 작성하라. 사용된 단어는 오름차순으로 정렬한다. 결과는 text.res2에 저장한다.

- 1) 저수준의 입출력을 사용
- 2) 고수준의 입출력을 사용
- 3) lex를 이용한 입출력

구현 과정 : 일단 구현하는 데 있어서 신경 써야 할 요소는 ‘어떻게 단어를 구별할 것인가’와 ‘어떻게 사용 횟수를 카운팅 할 것인가’이다. 우선 단어 구별은 한 바이트씩 읽어오면서 그 문자가 알파벳인지와 아닌지를 구별하였으며, 알파벳이라면 char*형 버퍼에 저장하였다. 그렇게 읽어오다가 알파벳이 아닌 문자가 읽혔다면 버퍼 마지막에 널문자를 넣고 버퍼의 첫 문자가 널문자가 아니

라면 직접 구현한 연결 리스트에 삽입하였다.

연결 리스트에 대해 설명하자면, Seek()함수를 통해 버퍼의 문자열과 리스트 내부에 같은 이름을 가진 노드가 있다면 추가를 하지 않고 그 노드에 선언되어 있는 카운트를 1만큼 증가시켜주고 없다면 리스트에 추가를 시켜주는 방식이다. 물론 리스트에 노드를 추가하는 Insert()함수는 strcmp함수를 내부에서 이용해 자동으로 오름차순으로 추가가 된다.

<pre>a : 334 abandon : 1 aberration : 1 able : 1 about : 40 above : 3 absence : 1 absent : 1 absolutely : 3 absorbed : 2 abstracted : 1 abstraction : 1 abstruse : 1 abstruseness : 1 absurd : 1 accident : 6 accidents : 1 accompanied : 2 accompany : 3 accomplished : 2 according : 2 accordingly : 1 account : 1 accounting : 1 accumulations : 1 accuracy : 1 accurately : 1 accustomed : 1 achievement : 1 acquaintance : 1 acquainted : 1 act : 2 acting : 1 action : 2 actual : 1 actually : 2 adapted : 2 addition : 1 additional : 1 adjusting : 1 admission : 1 admitted : 2 admitting : 1 adopt : 1 adventure : 2 advice : 1 affair : 2 affairs : 1 afford : 1 afforded : 1 afloat : 1 afraid : 1 after : 12 afternoon : 1 afterwards : 6 again : 15 --More--</pre>	<pre>the_gold_bug.res2 x a : 334 abandon : 1 aberration : 1 able : 1 about : 40 above : 3 absence : 1 absent : 1 absolutely : 3 absorbed : 2 abstracted : 1 abstraction : 1 abstruse : 1 abstruseness : 1 absurd : 1 accident : 6 accidents : 1 accompanied : 2 accompany : 3 accomplished : 2 according : 2 accordingly : 1 account : 1 accounting : 1 accumulations : 1 accuracy : 1 accurately : 1 accustomed : 1 achievement : 1 acquaintance : 1 acquainted : 1 act : 2 acting : 1 action : 2 actual : 1 actually : 2 adapted : 2 addition : 1 additional : 1 adjusting : 1 admission : 1 admitted : 2 admitting : 1 adopt : 1 adventure : 2 advice : 1 affair : 2 affairs : 1 afford : 1 afforded : 1 afloat : 1 afraid : 1 after : 12</pre>
---	--

