2주차 결과보고서

전공: 경제학과 학년: 4학년 학번: 20180501 이름: 김연수

1. fmt 구현 알고리즘
2. main

main 함수

1. Remove\_Blanks\_At\_The\_End:

Remove\_Blanks\_At\_The\_End 함수는 전달받은 문자열의 끝에 있는 공백 문자를 삭제하는 함수이다. 함수 내부에서는 줄바꿈이 있는지, '\0'이 있는지를 확인하며, 전체 문자열의 끝에서부터 거꾸로 탐색하여 공백이 아닌 문자가 나오면 해당 위치까지 문자열을 잘라내어 반환한다.

1. Get\_Blanks\_Chars

Get\_Blanks\_Chars 함수는 전달받은 문자열의 시작 위치와 함께 공백 문자의 수와 문자의 수를 계산하는 함수이다. 함수 내부에서는 문자열을 순차적으로 탐색하며, 공백 문자가 나올 때마다 이전에 공백 문자가 있었는지를 확인하여 공백 문자의 수와 문자의 수를 계산한다. 문자열의 끝이나 줄바꿈 문자가 나오면 탐색을 멈추고 계산된 값을 반환한다.

1. Print\_Line

Print\_Line 함수는 문자열과 출력된 문자의 수, 이전 줄이 끝났는지 여부를 입력값으로 받아 처리하는 함수이다. 이 함수는 Get\_Blanks\_Chars 함수를 호출하여 공백 문자와 일반 문자의 수를 계산하고, 이전 줄과 합칠 수 있는 경우, 현재 줄에 추가할 수 있는 경우, 새로운 줄에 출력해야 하는 경우에 따라 문자열을 처리한다. 마지막으로, 문자열이 모두 출력될 때까지 while 루프를 반복한다.

1. 테스트

fmt결과화면 :

텍스트이(가) 표시된 사진

자동 생성된 설명

Header.h 파일에서 LEMIT을 40으로 설정했다. 그리고 fmt를 실행한 결과화면이다. fmt에 LEMIT 40이 잘 적용됐다. 텍스트의 길이가 40으로 제한되어 출력되는 것을 확인할 수 있었다.

1. 실습 결과레포트 :

3-1: 결과 화면 첨부

실습 1 결과 화면 첨부 :

텍스트이(가) 표시된 사진

자동 생성된 설명실습1 메인 캡처

dog.c , cat.c, cow.c는 각각 dog, cat, cow를 출력하는 함수이다. 여기서는 test.c 가 main.c의 역할을 한다.

실행화면 :

텍스트이(가) 표시된 사진

자동 생성된 설명

결과로 cat, cow, dog가 제대로 출력되는 것을 확인할 수 있었다.

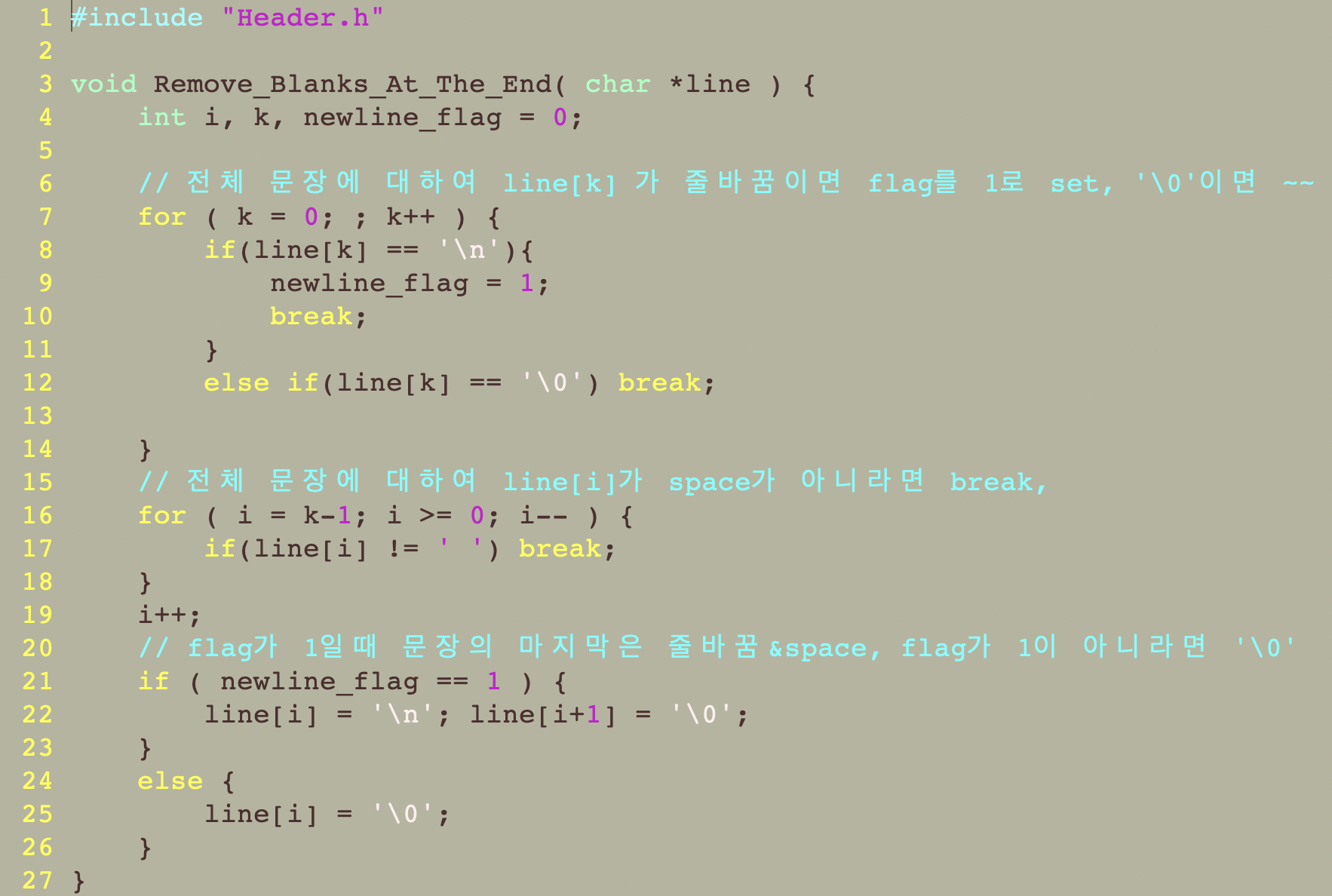
실습 2 결과 화면 첨부 :

텍스트이(가) 표시된 사진

자동 생성된 설명

gdb 캡처 맨 아래부분을 보면 i=1일때 “num=i/2 + I”의 연산 결과로 num가 1이 되는 것을 확인할 수 있다. 이는 i가 int자료형이기 때문에 발생한 일이다. i/2의 연산 결과로 0.5가 도출되어야 하는데, int자료형이기 때문에 연산 결과가 0이 되어버린다. 따라서 num = 1이 출력되는 것이다. 그렇기 때문에 i를 double자료형으로 변환시켜 문제를 해결해주었다. 해결한 결과 이 경우에 num=1.5로 출력되는 것을 확인할 수 있었다.

실습 3 결과 화면 첨부 :



line 7 ~ 12 : 한 글자씩 차례대로 읽고, 줄바꿈이면 flag를 1로 설정 후 break. ‘\0’이라면 그냥 break

line 16 ~ 19 : 다시 앞으로 line을 읽으면서 공백이 아닐 때까지 위치로 이동. 그리고 break. i++로 다시 뒤로 한칸 이동.

line 21 ~ 26 : flag가 1이라면 줄바꿈, flag가 1이 아니라면 ‘\0’를 넣어준다.

3-2. fmt를 구현하기 사용된 함수

1. main

main 함수

1. Remove\_Blanks\_At\_The\_End:

Remove\_Blanks\_At\_The\_End 함수는 전달받은 문자열의 끝에 있는 공백 문자를 삭제하는 함수이다. 함수 내부에서는 줄바꿈이 있는지, '\0'이 있는지를 확인하며, 전체 문자열의 끝에서부터 거꾸로 탐색하여 공백이 아닌 문자가 나오면 해당 위치까지 문자열을 잘라내어 반환한다.

1. Get\_Blanks\_Chars

Get\_Blanks\_Chars 함수는 전달받은 문자열의 시작 위치와 함께 공백 문자의 수와 문자의 수를 계산하는 함수이다. 함수 내부에서는 문자열을 순차적으로 탐색하며, 공백 문자가 나올 때마다 이전에 공백 문자가 있었는지를 확인하여 공백 문자의 수와 문자의 수를 계산한다. 문자열의 끝이나 줄바꿈 문자가 나오면 탐색을 멈추고 계산된 값을 반환한다.

1. Print\_Line

Print\_Line 함수는 문자열과 출력된 문자의 수, 이전 줄이 끝났는지 여부를 입력값으로 받아 처리하는 함수이다. 이 함수는 Get\_Blanks\_Chars 함수를 호출하여 공백 문자와 일반 문자의 수를 계산하고, 이전 줄과 합칠 수 있는 경우, 현재 줄에 추가할 수 있는 경우, 새로운 줄에 출력해야 하는 경우에 따라 문자열을 처리한다. 마지막으로, 문자열이 모두 출력될 때까지 while 루프를 반복한다.

3-3. 실습 makefile :

텍스트이(가) 표시된 사진

자동 생성된 설명

1 line : gcc를 사용하여 컴파일 한다.

2 line : 컴파일 옵션은 디버깅 가능하도록 하는 ‘-g’

3 line : 완성파일의 이름은 execute

5 line : objects에는 함께 컴파일할 오브젝트 파일 목록을 적어줬다.

7 line : 룰. target ‘execute’로 object파일들을 함께 컴파일할 것이다.

8 line : 명령. 해석하면 gcc -g -o execute dog.o cat.o cow.o test.o이다. -g는 디버깅을 하기위해 필요한 옵션이다. -o는 타겟파일의 이름을 execute로 지정하기 위해 필요하다.

9 ~ 19 line : 각 오브젝트 파일을 생성하기 위한 명령이다. ‘-c’ 옵션은 linking 이전 과정만 수행하도록 한다.

3-4. R5 규칙

텍스트이(가) 표시된 사진

자동 생성된 설명

규칙 R5: 입력줄의 첫 글자가 blank이면 앞줄과 합쳐지지 않게 한다. 만일, 줄의 첫 부분에 여러 개의 blank가 있으면 이 역시 줄을 바꾸어 새 줄에 출력하고 첫 부분의 blank는 첫 번째 blank를 포함해 그 개수만큼 그대로 출력한다.

규칙 R5를 구현하기 위한 알고리즘은 main.c 파일의 다음과 같다.

텍스트, 장치, 블랙, 계량기이(가) 표시된 사진

자동 생성된 설명

입력 줄의 첫 글자가 ‘ ‘이고 count가 0이 아닌 경우(빈 라인이 아닌 경우) \n을 출력하고, 다음 라인에서 시작해야 하므로 count = 0; B\_Flag = 0; 을 해준다. 그리고 line1에 line2를 넣어준 후에 다음 루프에서 print\_line을 통해 출력해준다.

3-5. 주로 많이 쓰는 make 옵션은 다음과 같다.

1. -f FILE, --file=FILE, --makefile=FILE: 파일을 메이크파일로 읽는다.

2. -j [N], --jobs[=N]: 한 번에 N개의 작업을 수행한다. 인수를 지정하지 않으면 무제한으로 작업한다.

3. -n, --just-print, --dry-run, --recon: 실제로 레시피를 실행하지 않고 출력만 한다.

4. -C DIRECTORY, --directory=DIRECTORY: 작업을 수행하기 전에 디렉토리를 변경한다.

5. -B, --always-make: 모든 대상을 무조건적으로 다시 만든다.

6. -k, --keep-going: 몇몇 대상이 만들어지지 않아도 계속해서 작업한다.

7. -s, --silent, --quiet: 레시피를 출력하지 않는다.

8. -i, --ignore-errors: 레시피 실행 중 오류를 무시한다.

9. -p, --print-data-base: 내부 데이터베이스를 출력한다.

10. -t, --touch: 대상을 다시 만들지 않고 최신 상태로 만든다.