2주차 결과보고서

전공: 수학/컴퓨터공학 학년: 3학년 학번: 20181288 이름: 윤성호

1. [사진 1], [사진 2], [사진 3]은 각각 LIMIT 값을 72, 20, 36을 설정하고 pdf파일의 예시 텍스트를 출력한 것이다.

텍스트이(가) 표시된 사진

자동 생성된 설명

[사진 1]

텍스트이(가) 표시된 사진

자동 생성된 설명

[사진 2]

텍스트이(가) 표시된 사진

자동 생성된 설명[사진 3]

2. fmt를 구현하기 위해서 여러개의 c파일을 만들었다. 전체적인 흐름을 다루는 main.c, 전처리과정과 문장의 시작공백과 단어를 구성하는 문자의 수를 알아내는 String\_Manipulation.c, 텍스트파일을 원하는 형식으로 출력하도록 도와주는 Output.c로 구성되어 있고, c파일 안에 들어있는 함수들과 #define 값들은 Header.h라는 파일에 정의해 놓았다.

main.c

텍스트파일을 입력받고 fmt 명령에 맞게 텍스트를 출력해준다. 텍스트파일은 꼭 하나만 입력을 집어넣어야하며 그렇게 하지 않으면 “Usage: fmt filename | > outfile” 이라는 에러메세지를 띄워준다. 그 이후 파일을 입력받은 후, line1을 fgets 함수를 이용해 빋아온다. 이때 아무것도 받아오지 못했다면 파일이 비어있다고 판단할 수 있고 함수를 종료해준다. 잘 받아왔다면 Remove\_Blanks\_At\_The\_End 함수를 호출하여 문자열 뒤에 공백을 없애주는 전처리를 진행해준다. 그 이후에는 Count(실제 출력한 글자들의 개수를 저장하는 변수), B\_Flag(Count != 0 이면 1로 지정, 다음줄이 빈줄이거나 첫 글자가 blank이면 0으로 지정), 그리고 B\_Line(전처리과정 후 그 줄이 빈줄이라면 1로 지정, 아니면 0으로 지정) 들을 이용하여 명령을 수행하고, line2를 받아온다. 그 line2가 R5 조건에 걸리면 다음줄에 출력하고 걸리지 않으면 line1 바로 옆에 출력한다. 그리고 같은 과정을 텍스트파일의 EOF가 검출될 때까지 진행한다. 마지막 줄이 빈줄인 경우 출력되지 않는 버그가 발생할 수 있는데 이는 while loop 종료 후 마지막 줄이 빈줄인지 아닌 지 체크해줘서 따로 출력을 해주는 과정을 추가해줬다.

String\_Manipulation.c

Remove\_Blanks\_At\_The\_End와 Get\_Blanks\_Chars가 들어가있다. Remove\_Blanks\_At\_The\_End 함수는 R1~R9의 조건에 부합하게 문자열을 출력하기 전에 그 문자열의 마지막 글자와 줄바꿈문자 사이의 공백을 지워주는 함수이다.

Get\_Blanks\_Chars 함수는 띄어쓰기가 포함된 여러개의 단어를 분석하는데, 단어앞쪽 공백개수와,단어의 철자개수를 받아온다. 결과적으로 문장단위가 아닌 단어단위로 분석을 하며 Print\_Line함수에서 계속 이어 붙일지 띄어서 새로운 줄에 쓸지를 매번 결정할 수 있도록 도와준다.

Output.c

Print\_Line 함수는 line을 받아와서 Get\_Blanks\_Chars에 의해 분석된 단어들을 R1~R9의 조건에 맞게 출력을 해주는 함수이다. 그 문장의 마지막 단어까지 출력을 하면 함수는 종료한다.

3.

<Makefile>

CC = gcc

TARGET = animal.exe # 만들어질 파일은 animal.exe이다.

OBJECTS = dog.o blackcow.o turtle.o main.o

# .o 파일이 없으면 .c파일을 자동으로 만들어준다.

$(TARGET) : $(OBJECTS)

#(CC) -o $(TARGET) $(OBJECTS)

# animal.h 파일이 바뀌어도 오브젝트 파일에 다 반영이 된다.

$(OBJECTS) : animal.h

# .PHONY 를 통해 clean이라는 이름의 파일이 있어도 make clean 명령어를 실행가능하게 만든다.

.PHONY : clean

clean:

rm $(OBJECTS) $(TARGET)

변수를 활용하여 Makefile의 가독성과 유지 보수를 더 편리하게 만들었다.

CC 는 gcc 컴파일러를 이용하겠다는 의미이다.

TARGET은 최종적인 실행파일의 이름이다.

OBJECTS는 만들어질 오브젝트 파일들의 이름들을 저장해놓는 변수이다.

TARGET을 목표로 하고 OBJECTS가 의존해있는 상태에서 recipe는 gcc 컴파일러를 이용해 object파일들을 가지고 target이란 이름의 실행파일을 생성하라는 의미이다.

오브젝트 파일에 대해 animal.h가 의존하고 있다는 의미이다.

.PHONY를 활용하면 clean이라는 파일명이 있더라도 make clean 명령어를 쓸 수 있게 해준다.

clean명령어를 통해 TARGET과 오브젝트파일을 모두 지우고, c언어 파일만 남길 수 있다.

4. R5는 입력줄의 첫 글자가 blank이면 앞줄과 합쳐지지 않게 줄바꿈을 하라는 규칙이다. 이럴 때는 상황이 두가지이다. 첫번째는 Count가 0이 아닌경우이다. Count가 0이 아니라는 것은 앞줄의 출력은 끝났지만 LIMIT는 아직 넘지 않아서 더 이어붙여서 출력이 가능할 수도 있다는 것을 말한다. 그러나 R5 규칙을 지키기 위해 putchar(‘\n’)로 줄바꿈을 해주어야 하며, 새로운 줄이므로 B\_Flag와 Count변수 모두 0으로 바꿔준다. 그 다음에 Print\_Line함수에서 공백을 포함하게 출력해준다. 두번째로 Count가 0인 경우이다. 이러한 상황에는 별다른 처리없이 바로 Print\_Line함수에 들어가서 공백을 포함해서 출력을 해주면 된다.

5.

Make에 옵션을 추가해주게 되면 매번 makefile에서 지정을 해주지 않고 터미널에서 원하는 정보들을 얻어올 수 있다.

아래는 몇가지의 대표적인 make 옵션이다.

-c dir : Makefile을 계속 읽지 말고 dir로 이동하라는 명령이다.

-d : (-debug) Makefile을 수행하면서 각종 정보를 모두 출력해준다.

-h : (-help) 옵션의 도움말을 출력해준다.

-r : 내장하고 있는 규칙들을 없는 것으로 간주하여 프로그램이 실행된다. 그래서사용자가 규칙을 새롭게 정의해주어야 한다.

-p : (-print-data-base) make에서 내부적으로 세팅되어 있는 값을 출력한다.

-v : (-version) make의 버전을 출력한다.

-t : (-touch)파일의 생성날짜를 현재 시간으로 업데이트한다.

-k : (-keep-going) make에서 에러가 나도 계속 진행하라는 명령이다.

-j : (job) 한번에 수행할 수 있는 명령을 지정하는 페럴렐옵션이다.