

CSE3013 (컴퓨터공학 설계 및 실험 I)

UNIX-2 결과 보고서

서강대학교 컴퓨터공학과 박수현 (20181634)

서강대학교 컴퓨터공학과

1 목적

실습 과정에 개발한 fmt에 대하여 결과 보고한다.

2 문제 풀이 결과

2.1 알고리즘

`int main(int argc, char *argv[]):` 메인 메서드.

1. FILE *fp, char *line1, char *line2, char *tmpline 정의.
2. int Count, int Blanks, int B_Flag, int B_Line 변수 0으로 초기화.
3. 파일 이름이 주어지지 않았을 경우: Usage: fmt filename | > outfile 출력 후 프로그램 종료.
4. 파일 읽기 시도. 파일 읽기에 실패할 경우: File open error. 출력 후 프로그램 종료.
5. line1, line2의 메모리 할당에 실패할 경우: Memory allocation error 출력 후 프로그램 종료.
6. 입력: line1
7. line1의 길이 = 0일 경우: 프로그램 정상 종료.
8. Remove_Blanks_At_The_End(line1).
9. line1의 첫 글자가 \n일 경우: \n 출력.
10. 루프:
 - i. B_Line이 0일 경우: Print_Line(line1, &Count, &B_Flag), 아닐 경우: B_Line \leftarrow 0.
 - ii. Count가 0이 아닐 경우: B_Flag \leftarrow 1.
 - iii. 입력: line2. 읽지 못했을 경우: 루프 탈출.
 - iv. Remove_Blanks_At_The_End(line2).
 - v. line2[0]이 공백 문자이거나 Count가 0이 아닐 경우:
 1. \n 출력.
 2. B_Flag \leftarrow 0, Count \leftarrow 0.
 - vi. 그렇지 않으면서 line2[0]이 개행 문자일 경우:

1. B_Flag이 1일 경우: \n 출력, B_Flag \leftarrow 0
2. \n 출력.
3. B_Line \leftarrow 1, Count \leftarrow 0.
- vii. line1과 line2의 내용 서로 바꾸기.
11. line1의 첫 글자가 개행 문자가 아닐 경우:
 - i. i 변수 0으로 초기화.
 - ii. 루프:
 1. line1[i]가 개행 문자 혹은 NUL 문자일 경우: 루프 탈출.
 2. i \leftarrow i + 1.
 - iii. line1[i]가 개행 문자일 경우: \n 출력.
12. 프로그램 정상 종료.

void Remove_Blanks_At_The_End(char *line): 문자열 뒤의 연속된 공백을 제거하는 메서드.

1. int i 정의 및 int k, int newline_flag 변수 0으로 초기화.
2. 루프:
 - i. line[k]이 개행 문자일 경우: newline_flag \leftarrow 1, 루프 탈출.
 - ii. line[k]이 NUL 문자일 경우: 루프 탈출.
 - iii. k \leftarrow k + 1.
3. i \leftarrow k - 1.
4. 루프:
 - i. line[i]가 공백 문자가 아닐 경우: 루프 탈출.
 - ii. i \leftarrow i - 1.
 - iii. i가 0보다 작을 경우: 루프 탈출.
5. newline_flag이 1일 경우:
 - i. line[i + 1] \leftarrow LF (\n).
 - ii. line[i + 2] \leftarrow NUL (\0).
6. newline_flag이 1이 아닐 경우:
 - i. line[i + 1] \leftarrow NUL (\0).
7. 프로시저 종료

void Get_Blanks_Chars(char *line, int Start, int *N_Blanks, int *N_Chars): 문자열의 공백과 문자를 세는 메서드.

1. int i 정의 및 int blank_flag 변수 0으로 초기화.
2. *N_Blanks \leftarrow 0, *N_Chars \leftarrow 0, i \leftarrow Start.
3. 루프:
 - i. line[i]이 개행 문자 혹은 NUL 문자일 경우: 루프 탈출.
 - ii. line[i]이 공백 문자일 경우:
 1. blank_flag가 0일 경우: *N_Blanks \leftarrow *N_Blanks + 1.

- 2. blank_flag가 0이 아닐 경우: 루프 탈출.
- iii. line[i]이 개행 문자, NUL 문자, 공백 문자 중 하나가 아닐 경우:
 - 1. blank_flag \leftarrow 1
 - 2. *N_Chars \leftarrow *N_Chars + 1
- 4. 프로시저 종료

void Print_Line(char *line, int *Count, int *B_Flag): 문자열의 공백 수와 문자 수를 세는 메서드.

- 1. int i 정의 및 int k, int newline_flag 변수 0으로 초기화.
- 2. 루프:
 - i. line[k]이 개행 문자일 경우: newline_flag \leftarrow 1, 루프 탈출.
 - ii. line[k]이 NUL 문자일 경우: 루프 탈출.
 - iii. k \leftarrow k + 1.
- 3. i \leftarrow k - 1.
- 4. 루프:
 - i. line[i]가 공백 문자가 아닐 경우: 루프 탈출.
 - ii. i \leftarrow i - 1.
 - iii. i가 0보다 작을 경우: 루프 탈출.
- 5. newline_flag이 1일 경우:
 - i. line[i + 1] \leftarrow LF (\n).
 - ii. line[i + 2] \leftarrow NUL (\0).
- 6. newline_flag이 1이 아닐 경우:
 - i. line[i + 1] \leftarrow NUL (\0).
- 7. 프로시저 종료

void Get_Blanks_Chars(char *line, int Start, int *N_Blanks, int *N_Chars): 문자열을 설정된 길이에 맞게 포맷하여 출력하는 메서드.

- 1. int i, int N_Blanks, int N_Chars 정의 및 int Start 변수 0으로 초기화.
- 2. Get_Blanks_Chars(line, Start, &N_Blanks, &N_Chars).
- 3. 루프:
 - i. *B_Flag가 1이면서 *Count + N_Chars + 1 \leq LIMIT인 경우:
 - 1. N_Blanks가 0이 아닌 경우: Something Wrong! 출력 후 프로그램 에러 코드 -1로 종료.
 - 2. 공백 문자 1개 출력.
 - 3. *B_Flag \leftarrow 0.
 - 4. i \leftarrow Start.
 - 5. 루프:
 - 1. line[i] 출력.
 - 2. i \leftarrow i + 1.

3. $i \geq \text{Start} + \text{N_Chars}$ 일 경우: 루프 탈출.
6. $\text{Start} \leftarrow \text{Start} + \text{N_Chars}$, $\text{*Count} \leftarrow \text{*Count} + \text{N_Chars} + 1$.
7. $\text{Get_Blanks_Chars}(\text{line}, \text{Start}, \&\text{N_Blanks}, \&\text{N_Chars})$.
- ii. 3.1.이 아니면서, $\text{*Count} + \text{N_Blanks} + \text{N_Chars} \leq \text{LIMIT}$ 인 경우:
 1. $i \leftarrow \text{Start}$.
 2. 루프:
 1. $\text{line}[i]$ 출력.
 2. $i \leftarrow i + 1$.
 3. $i \geq \text{Start} + \text{N_Blanks} + \text{N_Chars}$ 일 경우: 루프 탈출.
 3. $\text{Start} \leftarrow \text{Start} + \text{N_Blanks} + \text{N_Chars}$, $\text{*Count} \leftarrow \text{*Count} + \text{N_Blanks} + \text{N_Chars} + 1$.
 4. $\text{Get_Blanks_Chars}(\text{line}, \text{Start}, \&\text{N_Blanks}, \&\text{N_Chars})$.
- iii. 3.1., 3.2가 아닌 경우:
 1. *Count 가 0인 경우:
 1. $i \leftarrow \text{Start}$.
 2. 루프:
 1. $\text{line}[i]$ 출력.
 2. $i \leftarrow i + 1$.
 3. $i \geq \text{Start} + \text{N_Blanks} + \text{N_Chars}$ 일 경우: 루프 탈출.
 3. $\text{Start} \leftarrow \text{Start} + \text{N_Blanks} + \text{N_Chars}$.
 4. $\text{Get_Blanks_Chars}(\text{line}, \text{Start}, \&\text{N_Blanks}, \&\text{N_Chars})$.
 5. $\text{Start} \leftarrow \text{Start} + \text{N_Blanks}$.
 2. *Count 가 0이 아닌 경우:
 3. *Count 가 0인 경우:
 1. 개행 문자 출력.
 2. $\text{*B_Flag} \leftarrow 0$.
 - 3.
 4. $i \leftarrow \text{Start} + \text{N_Blanks}$.
 5. 루프:
 1. $\text{line}[i]$ 출력.
 2. $i \leftarrow i + 1$.
 3. $i \geq \text{Start} + \text{N_Blanks} + \text{N_Chars}$ 일 경우: 루프 탈출.
 6. $\text{Start} \leftarrow \text{Start} + \text{N_Blanks} + \text{N_Chars}$.
 7. $\text{*Count} \leftarrow \text{N_Chars}$.
 8. $\text{Get_Blanks_Chars}(\text{line}, \text{Start}, \&\text{N_Blanks}, \&\text{N_Chars})$.
4. 프로시저 종료