「Korea Polytechnic University」
프로그래밍 과제 노트

2019-01학기

교과목 담당교수	컴퓨터공학부 박정민
학과	컴퓨터공학과
학번	2017152049
이름	정하림

프로그래밍 과제 노트 목차

1. 온라인 강의 요약정리

1-1 Part1-6장. 반복문

■ 반복문이란?

- 정해진 문장을 반복하여 실행하도록 해주는 제어문이다.
- 효율적으로 많은 양의 데이터를 처리할수 있게 해준다.
- C언어에서 for문과 while문 do~ while문이라는 세 가지 종류가 있다.

■ 반복문을 만드는 방법 1 - while문

-
```
while (조건식)
{
        실행문;
}
```

- 특정 조건의 만족 여부로 명령을 반복시킬 때 사용한다.
- 조건식이 참(!=0)일 경우 조건식 아래의 문장이나 { }(중괄호)연산자로 묶인 문단을 반복한 후 조건식을 확인한다. 만약 조건식이 거짓(0)일 경우 아래의 문장이나 문단을 실행시키지 않는다.
- 초기식, 조건식, 증강식이 정형화 되어있지 않아 자유로운 형태로 내용을 반복시킬 수 있다.
- 주로 실행문 내에 증강식을 통해 조건식에 영향을 주어 사용한다.
- 무한 반복시키고 싶을 때는 조건식에 1(무조건 참)을 넣는다. 이 경우에는 조건문과 break를 사용하여 무한루프에서 탈출시킬 수 있다.
- 중첩사용으로 더 복잡한 문제를 해결할 수 있다.

■ 반복문을 만드는 방법 2 - for문

-
```
for (초기식; 조건식; 증감식)
{
        실행문;
}
```

- while문과 동일하게 반복을 위해 사용하지만 작성방법이 그보다 더 고정되어있다.
- 초기식은 단 한번만 수행하고 불필요시 생략할 수 있다.
- 조건식은 반복여부를 결정해준다. 조건식이 참일 경우 실행문이 실행되고 증감식을 실행하고 다시 조건식을 본다. 조건식이 거짓이면 반복문이 종료되고 생략하면 항상 참이 된다.
- 증감식은 조건식에 변화를 주기위해 사용하며 이 또한 생략할 수 있다.
- 식이 얼마나 반복될지의 계수기 기본형식 내에 포함하고 있다.
- 초기식, 증감식을 쉼표를 통하여 여러 개 사용할 수도 있다.
- 중첩 사용으로 다차원 배열같은 문제들을 더 쉽게 해결할 수 있고 중첩 사용 시 제어변수를 다르게 해주는 것이 좋다.

■ 반복문을 만드는 방법 3 - do ~ while문

-
```
do
{
        실행문;
} while (조건식) ;
```

- while문과 사용방법이 동일하지만 조건식을 비교하기전 먼저 do 아래의 실행문을 실행하고 조건을 판단한다.
- 따라서 조건과 관계없이 최소 한번 실행하는 반복문을 만들 때 사용한다.
- 조건식의 뒤에 꼭 문장의 종료를 의미하는 세미콜론(;)을 넣어주어야 한다.

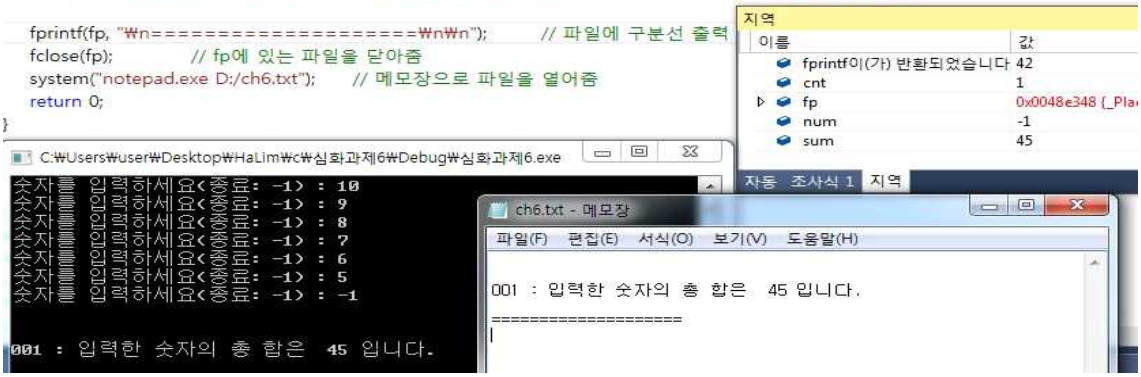
※ 전에 배운 break문과 continue문을 사용하여 무한 반복문을 끝내거나 반복문의 일부를 건너뛰고 다시 반복할 수 있다.

2. 심화문제 중점 문제 분석, 디버깅

2-1 심화 분석

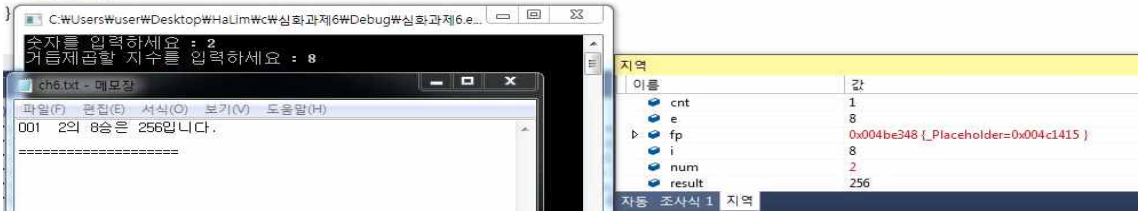
■ **심화 1번 문제 숫자를 계속 입력받고 -1을 입력받으면 그 전에 입력했던 수들의 최종 합을 출력해주는 프로그램을 작성하세요.**

```c
#pragma warning(disable:4996)
#include<stdio.h>

int main() {
    FILE * fp;                      // 파일 입출력을 위해 파일형 포인터 fp 선언
    fp = fopen("D:/ch6.txt", "at+");    // fp에 파일의 위치와 이어쓰기옵션을 넣어 파일을 열어줌
    int num = 0, sum = 0, cnt = 0;  // 숫자를 입력받을 num, 총합을 저장할 sum, 레코드 번호 cnt 선언

    while (num != -1) {             // num에 -1이 입력될때 까지 반복
        sum += num;                 // 먼저 sum에 일단은 0 인 num을 더해줌(-1을 계산하지 않기 위해)
        fprintf(stdout, "숫자를 입력하세요(종료: -1) : ");
        scanf("%d", &num);          // 숫자를 입력받음
    }
    fprintf(stdout, "₩n₩n%03d : 입력한 숫자의 총 합은  %d 입니다.₩n", ++cnt, sum);  // 화면에 결과 출력
    fprintf(fp, "₩n₩n%03d : 입력한 숫자의 총 합은  %d 입니다.₩n", cnt, sum);        //파일에 결과 출력

    fprintf(fp, "₩n====================₩n₩n");      // 파일에 구분선 출력
    fclose(fp);         // fp에 있는 파일을 닫아줌
    system("notepad.exe D:/ch6.txt");   // 메모장으로 파일을 열어줌
    return 0;
}
```



■ **심화 3번 문제**(사진크기 때문에 순서 변경)
숫자를 입력받아 반복문을 사용해 거듭제곱을 구하는 프로그램을 작성하세요.

```c
#pragma warning(disable:4996)
#include <stdio.h>

int main() {
    FILE * fp;                      // 파일 입출력을 위해 파일형 포인터 fp 선언
    fp = fopen("D:/ch6.txt", "at+");    // fp에 파일의 위치와 이어쓰기옵션을 넣어 파일을 열어줌
    int num, i, e, result, cnt = 0;
    // 입력값, 거듭제곱할 지수, 반복문에 사용할 변수, 결과값을 넣을 변수, 파일 레코드를 담을 변수 선언
    fprintf(stdout, "숫자를 입력하세요 : ");
    scanf("%d", &num);              // 사용자로부터 숫자를 입력받음
    fprintf(stdout, "거듭제곱할 지수를 입력하세요 : ");
    scanf("%d", &e);                // 거듭제곱할 지수 입력받음
    result = num;                   // 결과값을 입력값으로 초기화
    for (i = 1; i < e; i++)         // i에 1을 넣고 지수값의 거듭제곱 횟수에 따라 반복시킴
        result *= num;              // 반복마다 결과값을 제곱시킴
    fprintf(fp, "%03d  ", ++cnt);   // 파일에 레코드번호 출력
    fprintf(stdout, "%d의 %d승은 %d입니다.₩n",num,e,result);      // 화면에 결과 출력
    fprintf(fp, "%d의 %d승은 %d입니다.₩n", num, e, result);       // 파일에 결과 출력
    fprintf(fp, "₩n====================₩n₩n");      // 파일에 구분선 출력
    fclose(fp);         // fp에 있는 파일을 닫아줌
    system("notepad.exe D:/ch6.txt");   // 메모장으로 파일을 열어줌
    return 0;
}
```



■ **심화 2번문제) 숫자를 입력받아 입력받은 숫자로부터 하나씩 개수가 줄어드는 별(*)을 오른쪽 정렬로 출력해주는 프로그램을 작성하세요.**

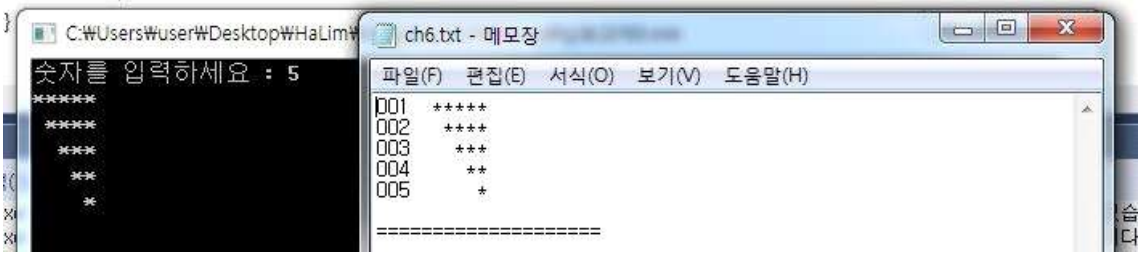
```
#pragma warning(disable:4996)
#include <stdio.h>
int main() {
    FILE * fp;                          // 파일 입출력을 위해 파일형 포인터 fp 선언
    fp = fopen("D:/ch6.txt", "at+");    // fp에 파일의 위치와 이어쓰기옵션을 넣어 파일을 열어줌
    int input, i, j, cnt=0;             // 숫자를 입력받을 input, 반복문에 사용할 i,j 레코드번호 cnt 변수 선언

    fprintf(stdout, "숫자를 입력하세요 : ");
    scanf("%d", &input);                // 숫자를 입력받음

    for (i=input; i > 0; i--) {         // 열의 갯수를 담당할 반복문
        fprintf(fp, "%03d  ", ++cnt);   // 파일에 레코드번호 출력
        for (j = input - i; j > 0; j--){    // 띄어쓰기를 담당할 반복문
            fprintf(stdout, " ");       // 화면에 띄어쓰기 출력
            fprintf(fp, " ");           // 파일에 띄어쓰기 출력
        }
        for (j = 0; j < i; j++) {       // "*" 찍기를 담당할 밤복문
            fprintf(stdout,"*");        // 화면에 별 출력
            fprintf(fp, "*");           // 파일에 별 출력
        }
        fprintf(stdout, "₩n");          // 화면에 개행문자 출력
        fprintf(fp, "₩n");              // 파일에 개행문자 출력
    } 경과 시간 21ms 이하
    fprintf(fp, "₩n====================₩n₩n");      // 파일에 구분선 출력
    fclose(fp);             // fp에 있는 파일을 닫아줌
    system("notepad.exe D:/ch6.txt");     // 메모장으로 파일을 열어줌
    return 0;
}
```

지역

이름	값
fprintf이(가) 반환되었습니다	1
cnt	5
fp	0x00
i	1
input	5
j	1

자동 조사식 1 지역

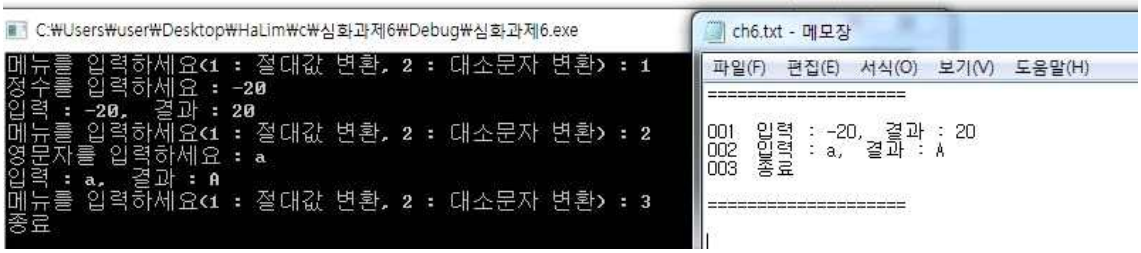
-



■ 심화 4번문제) 메뉴번호를 입력받아 조건에 맞게 출력하는 프로그램을 작성하시오.

```
#pragma warning(disable:4996)
#include <stdio.h>
int main() {
    FILE * fp;                          // 파일 입출력을 위해 파일형 포인터 fp 선언
    fp = fopen("D:/ch6.txt", "at+");    // fp에 파일의 위치와 이어쓰기옵션을 넣어 파일을 열어줌
    int menu, num, iResult= 0, cnt = 0;
    // 사용자에게 메뉴를 선택받을 menu, 절대값으로 바꿀 정수를 입력받을 num,
    // num의 절대값이 들어갈 iResult, 파일의 레코드번호 cnt 선언
    char a=0, cResult = 0, flag = 1;
    // 대소문자로 변환할 알파벳 a, 변환 후의 결과 cResult, 무한반복문 탈출을 위한 flag 선언

    while (flag) {                      // flag의 상태가 바뀔때 까지 무한으로 반복할 반복문
        fprintf(stdout, "메뉴를 입력하세요(1 : 절대값 변환, 2 : 대소문자 변환) : ");
        scanf("%d", &menu);             // 사용자에게 메뉴를 입력받음
        switch (menu) {                 // 사용자가 입력한 조건에 맞춰 각각의 명령 실행
        case 1 :
            fprintf(stdout, "정수를 입력하세요 : ");
            scanf("%d", &num);          // num에 정수를 입력받음
            if (num < 0) iResult = num*(-1);       //음수이면 양수로 바꿔줌
            else iResult = num;         // 양수일때는 그대로 결과값에 넣어줌
            fprintf(stdout, "입력 : %d,  결과 : %d₩n",num,iResult);   // 화면에 결과 출력
            fprintf(fp, "%03d  ", ++cnt);           // 파일 레코드 번호 출력
            fprintf(fp, "입력 : %d,  결과 : %d₩n", num, iResult);      // 파일에 결과 출력
            break;                      // 첫번째 케이스를 빠져나감

        case 2 :
            fprintf(stdout, "영문자를 입력하세요 : ");
            scanf(" %c", &a);               // 사용자에게 알파벳을 입력받음
            if (a > 64 && a < 91)       // 아스키코드 대문자범위인 65~90 일때
                cResult = a + 32;           // 대소문자의 차이인 32를 이용해 대문자를 소문자로 변환
            else if (a > 96 && a < 123)     // 아스키코드 소문자범위인 97~122 일때
                cResult = a - 32;           // 소문자를 대문자로 변환
            else {                          // 입력이 알파벳 대문자, 소문자도 아닐때 예외처리
                fprintf(stdout, "영문자를 입력해주세요₩n");     // 화면에 안내문구 출력
                fprintf(fp, "%03d  ", ++cnt);                   // 파일 레코드 번호 출력
                fprintf(fp, "영문자를 입력해주세요₩n");          // 파일에 안내문구 출력
                continue;                   // 결과를 출력하지 않고 다시 while문으로 돌아가게함
            }
            fprintf(stdout, "입력 : %c,  결과 : %c₩n", a, cResult);      // 화면에 결과 출력
            fprintf(fp, "%03d  ", ++cnt);                                // 파일 레코드 번호 출력
            fprintf(fp, "입력 : %c,  결과 : %c₩n", a, cResult);          // 파일에 결과 출력
            break;
        default :                       // 1도 2도 아닌 다른 수가 입력되었을 때 예외처리
            fprintf(stdout, "종료₩n");      // 화면에 종료문구 출력
            fprintf(fp, "%03d  ", ++cnt);   // 파일 레코드 번호 출력
            fprintf(fp, "종료₩n");          // 파일에 종료문구 출력
            flag = 0;                       // while 무한루프를 탈출하기위해 flag를 거짓으로 변경
            break;
        }
```

```
    }
    fprintf(fp, "₩n====================₩n₩n");          // 파일에 구분선 출력
    fclose(fp);                                  // fp에 파일을 닫아줌
    system("notepad.exe D:/ch6.txt");            // 메모장으로 파일을 열어줌
    return 0;
}
```

- 

- 

- 

3. 자기성찰 및 평가

3-1 5주차-자기성찰 및 평가

■ 수업 및 실습을 통해서 배운 내용

- 반복문의 정의
- while의 정의와 활용법
- for문의 정의와 활용법
- do while문의 정의와 활용법
- 중첩 반복문을 만들고 활용하는 법
- 파일 입출력을 활용하여 메모장에 출력결과 넣는 방법

■ 느낀점 (자유롭게 기술)

- 반복문을 통해 코드를 반복 기술할 필요가 없어 더욱 편해진 것 같은데 내용적으로 훨씬 어려워진 것 같다. 반복문을 중첩시킬 때 증감식 활용에서 생각할 부분이 많았고, 노트에 정리하며 풀어보니 훨씬 쉽게 답을 찾을 수 있었다.
- 파일 입출력으로 평소쓰던 printf가 아닌 fprintf를 사용해보았는데 인수하나만 더 입력하면 돼서 이 부분은 생각보다 어렵지 않고 재밌었다. 내가 짠 코드가 바로 메모장에 찍히고 확인할 수 있게 되어서 뭔가 명령프롬프트에 값만 찍힐 때 보다 많은 곳에 활용할 수 있을 것 같다.
- 파일입출력 때문인지 디버깅시에 가끔씩 메모장에 이상한 값이 들어가는 오류가 떴으나 다시 실행하면 해결되었다. 무엇 때문에 이런현상이 발생하는지 모르겠다.
- 4번 문제를 풀며 반복문에 char형 변수를 입력받았는데 자꾸 scanf가 무시되었다, 내가 입력한 코드문제인줄 알았는데 아무리 돌아봐도 문법에는 이상이 없는 것 같았다. 디버깅으로 찾아보니 char형 변수에 값에 \n값이 들어가고 있었고 인터넷에서 해답을 찾았다.
  버퍼가 비워지지 않아서 발생하는 현상으로 fflush함수를 쓰거나 scanf(" %c") 형태로 띄어쓰기를 한칸 해주면 해결되는 문제였다. 디버깅을 통해 문제를 찾아서 해결해보니 뿌듯했고 정말 유용한 기능이라는 생각이 들었다.