

과목 : 프로그래밍 입문(2분반)

담당 교수 : 이승용

담당 조교 : 김종경

Assignment 2

- | | |
|----------------|---------------|
| 1. 프로그램 개요 | 5. 프로그램 실행 방법 |
| 2. 알고리즘 | 6. 프로그램 실행 예제 |
| 3. 프로그램 코드 | 7. 토론 및 결론 |
| 4. 프로그램 이해를 위해 | |

1. 프로그램 개요

□ 목적

이번 프로그램의 목적은 printf(), scanf(), 그리고 function 등의 사용에 익숙해지고 Loop, Select 문을 사용하여 예러 C 구현에 있어서 발생할 수 있는 에러를 처리할 수 있는 능력을 향상시키고자 함에 있다.

□ 전체구조

이번 Assignment1에서는 두 가지 문제를 해결해야 한다.

● Problem1 : 온도 단위 변환

Problem1의 전체구조는 매우 간단하다. 변환 하고자 하는 온도를 입력받고 입력 받은 온도의 단위가 무엇인지 char 형으로 입력 받은 다음에 입력 받은 온도가 F(Fahrenheit)일 경우에는 C(Celsius)로 C(Celsius)일 경우에는 F(Fahrenheit)로 변환하면 된다. F(Fahrenheit)와 C(Celsius)의 관계식은 다음과 같다.

$$C = (F - 32.0) / 1.8$$
$$F = C \times 1.8 + 32.0$$

출력은 소숫점 이하 둘째 자리 수에서 반올림하고 입력은 실수형으로 입력을 받으며, 단위를 입력 받을 때에는 C, F의 대소문자를 구분하지 않되 C, F 이외의 값이 들어올 경우에는 에러메세지를 출력하고 반복해서 입력을 받도록 한다.

● Problem2 : 두 직선의 관계

Problem2에서는 간단한 Geometry를 구현한다. 주어진 4개의 점을 통해서 두 개의 직선을 구하고 구한 두 직선 사이의 관계를 알아보면 된다. 두 직선 사이의 관계는 총 4가지가 나올 수 있는데, 교차(intersection), 평행(parallel), 일치(coincidence), 수직(perpendicular)의 관계이다. 따라서 입력으로 먼저 4개의 점을 입력 받아서 처음의 두 점을 통해서 직선 L1을 구하고, 다음의 두 점을 통해서 나머지 두 점을 통해서 직선 L2를 구한다. 두 점을 통한 두 직선을 구하는 관계식은 다음과 같다.

직선의 식을 $x+ay+b=0$ 의 형태로 나타낼 때, a, b를 구하면 직선을 구할 수 있다. (x의 계수가 0 경우는 예외처리) 두 점을 $(x_1, y_1), (x_2, y_2)$ 라고 하면 a, b를 구하는 식은 다음과 같다.

$$a = \frac{x_2 - x_1}{y_1 - y_2}, b = \frac{x_2 y_1 - x_1 y_2}{y_2 - y_1}$$

이 식을 통해서 우리는 직선의 방정식을 구할 수 있다. 그리고 L1과 L2의 직선 식을 통해서 두 직선 사이의 관계식을 구할 수 있다. (두 직선이 교차할 경우에) 교점의 좌표를 (x, y)라고 하고 직선 L1을 $x + a_1y + b_1 = 0$, L2를 $x + a_2y + b_2 = 0$ 이라고 하면 x, y를 구하는 식은 다음과 같다.

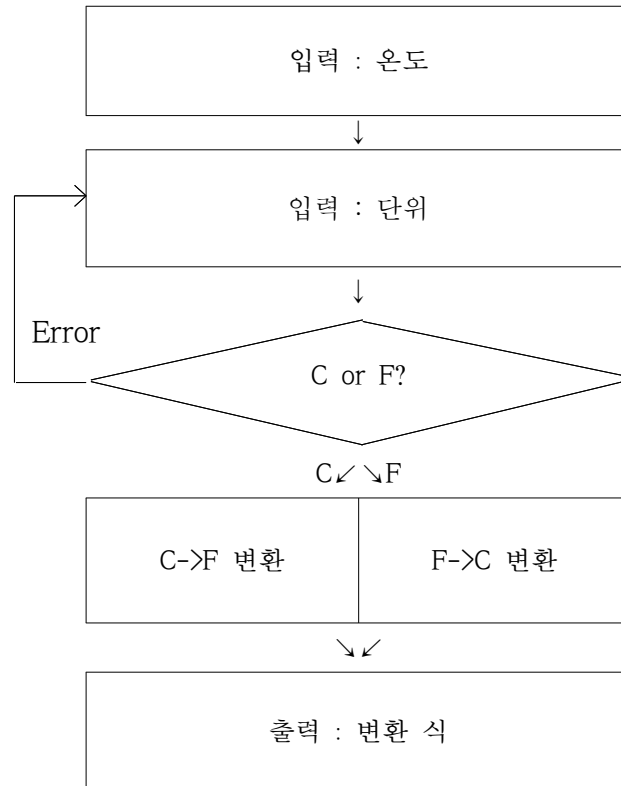
$$x = \frac{a_1b_2 - a_2b_1}{a_2 - a_1}, y = \frac{b_1 - b_2}{a_2 - a_1}$$

그리고 두 직선이 교차(intersect)하고 직교(perpendicular)하지 않을 때는 두 직선 사이의 각도를 예각으로 표시해 주어야 한다.

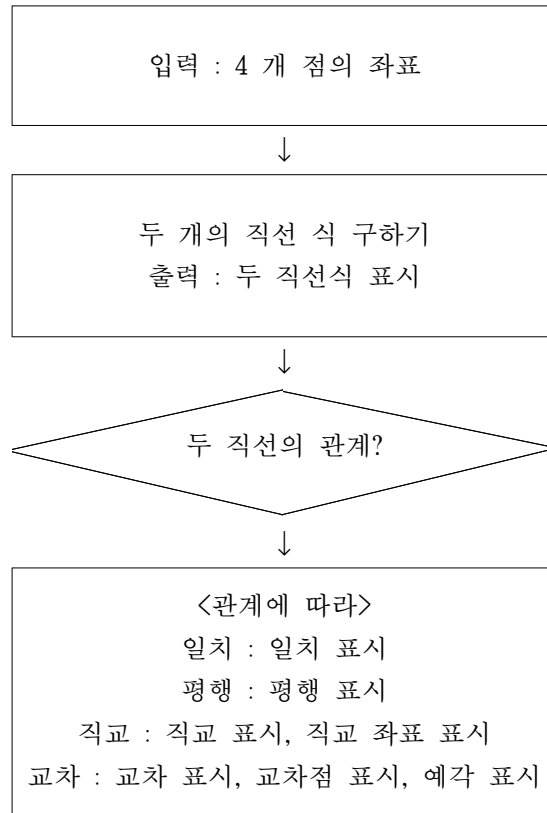
예외처리는 직선이 x축에 평행할 경우 처음 식에서 $y_2=y_1$ 이 되기 때문에 0으로 나눠주는 에러가 발생하는데 이를 처리해 주어야 하며, 두 직선이 평행할 경우 두 번째 식에서 $a_2=a_1$ 이 되기 때문에 이 역시 에러가 발생하여 이를 처리해 주어야 한다. 그리고 두 직선이 수직할 경우 다른 모든 경우는 arctan를 이용하여 판별이 가능하지만 두 직선이 각각 x축과 y축에 평행할 경우에는 arctan 값으로 구할 수 없기 때문에 이 경우에도 예외처리를 해주어야 한다.

2. 알고리즘

□ Problem1 : 온도 단위 변환



□ Problem2 : 두 직선의 관계



3. 프로그램 코드

□ Problem1 : 온도 단위 변환

● Header file 포함, Define, 함수 선언

```
#include <stdio.h>

double change_to_celsius(double temperature);
double change_to_fahrenheit(double temperature);
```

두 가지 사용자 정의 함수를 선언하였는데, 첫 번째 함수는 화씨로 입력 받은 온도를 섭씨온도로 바꿔주는 것이고 두 번째 함수는 섭씨로 입력 받은 온도를 화씨온도로 바꿔주는 함수이다.

● Main 함수, 단위 입력의 조건에 따라 처리하는 부분

```
while(1){
    printf("Is this a Fahrenheit or Celsius? [F/C] ");
    scanf("%c%c", &degree_type, &degree_type);
    curr_degree = degree_type;
```

```

        if(degree_type == 'c' || degree_type == 'C'){
            changed_tem = change_to_fahrenheit(temperature);
            degree_type = 'F';
            break;
        }
        else if(degree_type == 'f' || degree_type == 'F'){
            changed_tem = change_to_celsius(temperature);
            degree_type = 'C';
            break;
        }
        else{
            printf("Incorrect temperature scale!!!\n");
        }
    }

```

온도의 단위(degree_type)을 입력 받았을 때, 그 단위가 c(or C), f(or F)인지를 구분하고 그 값에 따라서 c(or C)일 경우에는 change_to_fahrenheit(temperature)를 Call하고 f(or F)일 경우에는 change_to_celsius(temperature)를 Call 한다. 그리고 그 외의 값이 입력되었을 경우에는 에러처리를 한다. 위의 코드를 보면 flushall()이라는 함수가 삽입되어 있는 것을 알 수 있는데, 이는 앞에서 친 Enter가 다음에 있는 scanf에서 char형으로 입력을 받을 때 이상한 값이 들어가서 두 번 scanf하는 에러를 처리하기 위해서 앞의 buffer에 있는 모든 값을 지워주고자 사용하였다.

● 사용자 정의 함수 : change_to_celsius

```

double change_to_celsius(double temperature){
    return (temperature-32)/1.8;
}

```

● 사용자 정의 함수 : change_to_fahrenheit

```

double change_to_fahrenheit(double temperature){
    return temperature*1.8+32;
}

```

□ Problem2 : 두 직선의 관계

● Header file 포함, Define, 함수 선언

```

#include <stdio.h>
#include <math.h>

#define DEGREE_FACTOR 57.295779

```

```
void get_points(int *p_x, int *p_y, int point_num);
void get_a_line(int x1, int y1, int x2, int y2, double *a, double *b, int line_num);
void relation_of_lines(double a1, double b1, double a2, double b2);
```

atan()함수를 사용하기 위해서 math.h header file을 삽입하였고, radian 값을 degree 값으로 변환하기 위해서 1 rad에 해당하는 degree 값을 define하였다. 그리고 사용자 정의 함수로는 첫 번째, 두 점을 입력 받는 함수와 두 번째, 두 개의 좌표가 주어질 때 직선의 방정식을 구하는 함수와 세 번째, 두 직선의 방정식이 주어질 때 두 직선 사이의 관계를 구하는 함수를 정의하였다.

● Main 함수, 에러처리(한 직선에 동일한 좌표를 입력하였을 경우)

```
while(1){
    get_points(&p1_x, &p1_y, 1);
    get_points(&p2_x, &p2_y, 2);
    get_points(&p3_x, &p3_y, 3);
    get_points(&p4_x, &p4_y, 4);
    printf("\n");

    if ((p1_x == p2_x && p1_y == p2_y) || (p3_x == p4_x && p3_y == p4_y))
        printf("Error! You have not to insert the same points on a line.\n\n");
    else
        break;
}
```

만약 한 직선 L1 또는 L2 상의 좌표를 동일하게 입력할 경우에는 한 점만으로는 직선을 구할 수 없으므로 에러처리를 한다. 즉 “Error! You have not to insert the same points on a line.”라는 에러메세지를 출력하면서 입력을 다시하도록 하는 것이다.

● Main 함수, 예외처리(직교 : 두 직선이 각각 x축과 y축에 평행할 때), 그 밖에 정의한 함수 call

```
if (p1_x == p2_x && p3_y == p4_y){
    printf("L1 : x + 0.00y + %.2f = 0\n", -(double)p1_x);
    printf("L2 : 0.00x + y + %.2f = 0\n", -(double)p3_y);
    printf("L1 and L2 are Perpendicular.\n");
    printf("Intersection Point P(x, y) : %.2f, %.2f\n", (double)p1_x, (double)p3_y);
}
else if (p1_y == p2_y && p3_x == p4_x){
    printf("L1 : 0.00x + y + %.2f = 0\n", -(double)p1_y);
    printf("L2 : x + 0.00y + %.2f = 0\n", -(double)p3_x);
    printf("L1 and L2 are Perpendicular.\n");
}
```

```

        printf("Intersection Point P(x, y) : %.2f, %.2f\n", (double)p3_x, (double)p1_y);
    }
    else {
        get_a_line(p1_x, p1_y, p2_x, p2_y, &a1, &b1, 1);
        get_a_line(p3_x, p3_y, p4_x, p4_y, &a2, &b2, 2);
        relation_of_lines(a1, b1, a2, b2);
    }
}

```

첫 번째와 두 번째 if는 두 직선이 각각 x축과 y축에 평행할 때 아래에 나오는 것처럼 사용자 정의 함수에서 사용한 방법(atan()함수를 이용해서 구한 방법)으로는 판별할 수 없기에 따로 예외처리를 해주었다. 세 번째 if는 입력받은 4좌표로부터 두 개의 직선 L1, L2를 구하는 것과 두 직선 사이에 관계를 구하는 함수를 Call한다.

● 사용자 정의 함수 : get_points

```

void get_points(int *p_x, int *p_y, int point_num){
    printf("Enter the value x and y of P%d: ", point_num);
    scanf("%d %d", p_x, p_y);
    return;
}

```

좌표를 입력 받고 출력한다.

● 사용자 정의 함수 : get_a_line

```

void get_a_line(int x1, int y1, int x2, int y2, double *a, double *b, int line_num){
    if (y1 == y2){
        *a = 1;
        *b = -y1;
        printf("L%d : 0.00x + %.2fy + %.2f = 0\n", line_num, *a, *b);
    }
    else {
        *a = ((double)(x2-x1))/((double)(y1-y2));
        *b = ((double)(x2*y1-x1*y2))/((double)(y2-y1));
        printf("L%d : x + %.2fy + %.2f = 0\n", line_num, *a, *b);
    }
    return;
}

```

직선의 식을 $x+ay+b=0$ 의 형태로 나타낼 때, a, b를 구하여 직선의 방정식을 구한다. x의 계수가 0 경우($y_1=y_2$ 일 경우)는 예외처리를 해준다. 두 점을 $(x_1, y_1), (x_2, y_2)$ 라고 하면 a, b를 구하는 식은 다음과 같다.

$$a = \frac{x_2 - x_1}{y_1 - y_2}, \quad b = \frac{x_2 y_1 - x_1 y_2}{y_2 - y_1}$$

● 사용자 정의 함수 : relation_of_lines

```
void relation_of_lines(double a1, double b1, double a2, double b2){
    double inter_x, inter_y;
    double degree;
    if (a1 == a2 && b1 == b2)
        printf("L1 and L2 are Coincident.\n");
    else if (a1 == a2)
        printf("L1 and L2 are Parrallel.\n");
    else if (a1 * a2 == -1){
        printf("L1 and L2 are Perpendicular.\n");
        inter_x = (a1 * b2 - a2 * b1) / (a2 - a1);
        inter_y = (b1 - b2) / (a2 - a1);
        printf("Intersection Point P(x, y) : %.2f, %.2f\n", inter_x, inter_y);
    }
    else{
        printf("L1 and L2 are Intersect.\n");
        inter_x = (a1 * b2 - a2 * b1) / (a2 - a1);
        inter_y = (b1 - b2)/(a2 - a1);
        printf("Intersection Point P(x, y) : %.2f, %.2f\n", inter_x, inter_y);
        degree = (atan(-1/a1) + atan(1/a2)) * DEGREE_FACTOR;
        if (degree < 0)
            degree = - degree;
        if (degree > 90)
            printf("Dergee : %.2f\n", 180 - degree);
        else
            printf("Dergee : %.2f\n", degree);
    }
    return;
}
```

두 직선을 방정식의 계수만 있으면 두 직선의 관계를 모두 정의할 수 있기 때문에 한 직선에 계수 두 개 (a, b)씩 두 직선의 총 4개의 계수를 입력으로 하는 사용자 정의 함수를 만들었다. 먼저, $a_1=a_2$, $b_1=b_2$ 이면 두 직선이 일치함을 보여주었고, $a_1=a_2$ 인데 $b_1=b_2$ 가 아니면 두 직선이 평행함을 보여주었고, $a_1*a_2=-1$ 일 경우에는 서로의 기울기($-1/a_1$, $-1/a_2$)의 곱이 -1 이기 때문에 수직임을 표시해주고 그 때의 교차점을 구하여 표시해 주었다. 교차점을 구하는 식은 교차점의 좌표를 (x, y)라고 했

을 때 $x = \frac{a_1b_2 - a_2b_1}{a_2 - a_1}$, $y = \frac{b_1 - b_2}{a_2 - a_1}$ 가 됨을 앞에서 보여주었다. 마지막으로 일치하지

도 않고, 평행하지도 않으며 수직도 아니면 일반적으로 교차하는 관계이니

$x = \frac{a_1b_2 - a_2b_1}{a_2 - a_1}$, $y = \frac{b_1 - b_2}{a_2 - a_1}$ 이 식을 이용하여 교차점을 구하였고, 또 그 때의 두 직

선 사이의 예각을 표시하기 위해서 math.h에 있는 atan()함수를 사용하여 두 직선을 기울기를 통해 예각을 구하였다. 예각은 $(\text{atan}(-1/a_1) + \text{atan}(1/a_2)) * \text{DEGREE_FACTOR}$ 의 식을 통해서 구할 수 있었는데, 기울기가 $-1/a_1$, $-1/a_2$ 니깐 각각의 기울기에 대한 atan값을 구하고 이 두 값을 빼 주면 두 직선 사이의 각을 구할 수 있다. 그런데 예각을 구해야 하기 때문에 그 결과 값이 음수가 나오면 양수로 바꿔주었고, 또 그 값이 90도보다 크면 180에서 빼주어서 예각을 구하였다. 이때 DEGREE_FACTOR의 역할은 atan함수를 이용하면 얻게 되는 값이 radian 값이기 때문에 이를 Degree값으로 바꿔주기 위함이다.

4. 프로그램 실행 방법

□ Problem1 : 온도 단위 변환

먼저 “Enter the degree you want to convert:” 라는 메시지가 뜨면 원하는 온도의 값(실수형)을 입력한다. 그리고 “Is this a Fahrenheit or Celsius? [F/C]” 라는 메시지가 뜨면 F(or f) 또는 C(or c)의 Character를 입력한다. 만약에 그 외의 값을 입력하였을 경우에는 “Incorrect temperature scale!!!”의 에러메세지와 함께 “Is this a Fahrenheit or Celsius? [F/C]” 가 반복해서 메시지가 출력될 것이다. F(or f) 또는 C(or c)의 값이 입력된 후에는 F일 경우에는 화씨온도에서 섭씨온도로 변환되고, C일 경우에는 섭씨온도에서 화씨온도로 변환된 식이 출력될 것이다.

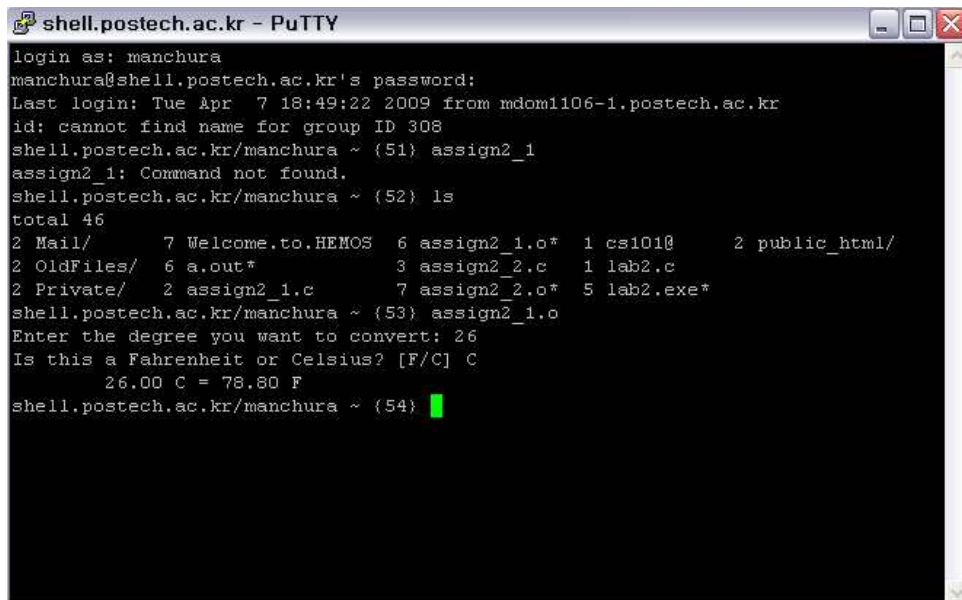
□ Problem2 : 두 직선의 관계

먼저 Enter the value x and y of P1: 라는 메시지가 뜨면 원하는 점의 좌표를 입력하는데 먼저 x좌표를 입력하고 Space bar로 공백을 준 뒤 다시 y좌표를 입력하고 Enter를 누른다. 계속해서 P2~P4까지의 좌표를 입력한다. 입력한 좌표 중 P1과 P2의 좌표는 한 직선 L1 상에 있는 좌표이고, P3과 P4의 좌표는 다른 한 직선 L2 상에 있는 좌표이다. 입력이 끝나면 입력 받은 좌표를 통해서 직선 L1과 L2의 직선 방정식이 출력되고, L1과 L2의 직선 사이의 관계(Coincidence, Parallel, Perpendicular, Intersect)가 결정되고, Perpendicular의 경우 교차점이 출력되고, Intersect의 경우 교차점과 두 직선 사이의 예각이 출력된다.

5. 프로그램 실행 예제

□ Problem1 : 온도 단위 변환

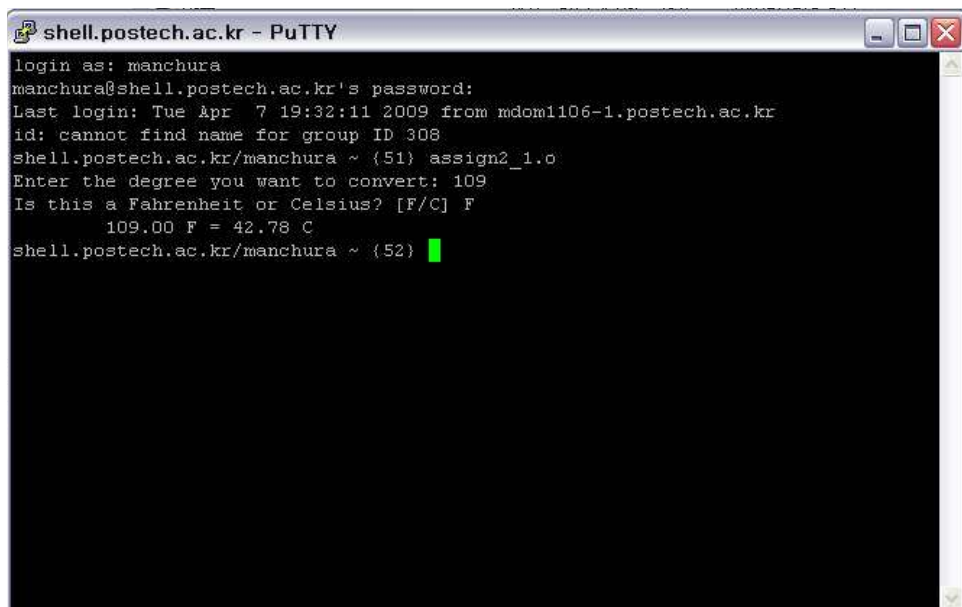
- 온도를 입력하고 단위로 C(or c)를 입력하였을 경우



```
shell.postech.ac.kr - PuTTY
login as: manchura
manchura@shell.postech.ac.kr's password:
Last login: Tue Apr 7 18:49:22 2009 from mdom1106-1.postech.ac.kr
id: cannot find name for group ID 308
shell.postech.ac.kr/manchura ~ {51} assign2_1
assign2_1: Command not found.
shell.postech.ac.kr/manchura ~ {52} ls
total 46
2 Mail/      7 Welcome.to.HEMOS  6 assign2_1.o*  1 cs1010      2 public_html/
2 OldFiles/  6 a.out*             3 assign2_2.c  1 lab2.c
2 Private/   2 assign2_1.c         7 assign2_2.o*  5 lab2.exe*
shell.postech.ac.kr/manchura ~ {53} assign2_1.o
Enter the degree you want to convert: 26
Is this a Fahrenheit or Celsius? [F/C] C
      26.00 C = 78.80 F
shell.postech.ac.kr/manchura ~ {54}
```

온도로 섭씨온도 26도를 입력하였더니 화씨온도 78.8F도로 변환되는 것을 확인할 수 있었다.

- 온도를 입력하고 단위로 F(or f)를 입력하였을 경우



```
shell.postech.ac.kr - PuTTY
login as: manchura
manchura@shell.postech.ac.kr's password:
Last login: Tue Apr 7 19:32:11 2009 from mdom1106-1.postech.ac.kr
id: cannot find name for group ID 308
shell.postech.ac.kr/manchura ~ {51} assign2_1.o
Enter the degree you want to convert: 109
Is this a Fahrenheit or Celsius? [F/C] F
      109.00 F = 42.78 C
shell.postech.ac.kr/manchura ~ {52}
```

화씨온도 109를 입력하였더니 섭씨온도 42.78로 변화되는 것을 확인할 수 있었다.

● 두 직선이 평행할 경우

[illegible]

● 두 직선이 수직할 경우

[illegible]

- [illegible]

- [illegible]

- L1이 x축에 평행할 경우

[illegible]

- L2가 y축에 평행할 경우

[illegible]

- 에러처리 : 한 직선 상에서 같은 좌표를 입력할 때

```

shell.postech.ac.kr - PuTTY
Error! You have not to insert the same points on a li
shell.postech.ac.kr/manchura ~ {89} assign2_2.o
Enter the value x and y of P1: 3 1
Enter the value x and y of P2: 3 1
Enter the value x and y of P3: 3 5
Enter the value x and y of P4: 5 6

Error! You have not to insert the same points on a line.

Enter the value x and y of P1: 4 2
Enter the value x and y of P2: 6 4
Enter the value x and y of P3: 4 5
Enter the value x and y of P4: 4 5

Error! You have not to insert the same points on a line.

Enter the value x and y of P1: 3 3
Enter the value x and y of P2: 3 3
Enter the value x and y of P3: 3 3
Enter the value x and y of P4: 3 3

Error! You have not to insert the same points on a line.

Enter the value x and y of P1: 

```

6. 토론 및 결론

□ Problem1 : 온도 단위 변환

온도 단위 변환 Program은 간단한 Warm up 프로그램으로써 그리 복잡하지 않았다. 단 조건을 만족시키기 위해서 사용자 정의 함수를 2개 이상 만들어야했고, 잘못된 입력에 대해서 에러처리를 해야 했다. 그런데 이번 Assign2에서 처음 char형 입력을 받았는데, char형 입력에 오류가 있다는 것을 알게 되었다. 그것은 앞에서 어떤 값을 입력받고 Enter가 입력될 때 다음에 scanf에서 %c형의 입력을 받을 경우 Enter를 입력으로 착각하는 것이었다. 그 때문에 한 번의 입력을 건너뛰게 된 것이다. 곰곰이 생각해본 결과 Enter의 입력을 받고 다시 원하는 입력을 받고자 %c를 두 번 입력받도록 설정하는 방법을 사용하였다. 이 문제 외에는 이번 Assign2의 Program1에서는 큰 문제가 없이 Coding을 할 수 있었다.

□ Problem2 : 두 직선의 관계

두 번째 문제는 두 직선 사이의 관계를 계산하는 문제였는데, 간단한 수학적 지식이 필요했다. 그 내용은 주어진 4개의 점을 통해서 두 개의 직선을 구하고 구한 두 직선 사이의 관계를 알아보는 것이었다. 앞에서 설명했듯이 두 점이 주어질 때 직선을 구하는 방법은 다음과 같다. 직선의 식을 $x+ay+b=0$ 의 형태로 나타낼 때, a, b를 구하면 직선을 구할 수 있는데, 이때 두 점을 $(x_1, y_1), (x_2, y_2)$ 라고 하면 a, b를

구하는 식은 다음과 같다. $a = \frac{x_2 - x_1}{y_1 - y_2}, b = \frac{x_2 y_1 - x_1 y_2}{y_2 - y_1}$ Coding에서는 이 과정을

get_a_line()이라는 사용자 정의 함수로 따로 만들어서 처리하였다. 입력으로 두 개의 좌표와 a, b(계수)의 주소가 보내지면 두 좌표 $(x_1, y_1), (x_2, y_2)$ 를 통해서 a, b의 주소에 위의 식을 통한 결과값을 저장하는 것이다. 그리고 또 하나의 함수를 더 만들었는데 그것은 두 직선 사이의 관계를 판별하는 relation_of_lines()라는 함수였다. 직선 사이의 관계는 계수들로 알 수 있다. 앞에서 y의 계수인 a값은 $-1/a$ 의 기울기 값을 나타냄을 알 수 있는데, 두 직선의 a값을 비교하여 평행한지 교차하는지 직교하는지를 판별하였다. 그리고 b값을 통해서 평행할 때 일치하는지를 또한 파악하였다. 직교하고 교차할 때는 교차점을 구해야 했는데, 교점의 좌표를 (x, y)라고 하고 직선 L1을 $x + a_1y + b_1 = 0$, L2를 $x + a_2y + b_2 = 0$ 이라고 하면 x, y를 구하는 식은 다음과 같다.

$$x = \frac{a_1b_2 - a_2b_1}{a_2 - a_1}, y = \frac{b_1 - b_2}{a_2 - a_1}$$
 그리고 교차할 때는 두 직선 사이의 예각

을 구해야 했는데 이는 inverse tangent 함수(atan)를 이용하여 그 차이를 구해주어서 구할 수 있었다. 그 밖에 많은 예외 처리가 있었는데, 프로그램 실행예제에서 그 예외들을 보여준 대로 한 직선 상에서 동일한 좌표를 입력하였을 때, 직선이 y축에 평행하거나 x축에 평행할 때, 그리고 직교할 경우 중에서 각각의 직선이 y축과 x축에 평행할 때는 모두 예외처리를 해주었다.

이번 Assign2의 Problem2는 수학적 지식을 활용해야하는 까다로움과 예외처리가 많았다는 번거로움이 어려움의 요소였고 그밖에 프로그래밍을 하는데는 큰 어려움이 없었다.