

|  |
| --- |
| **R E P O R T** |

|  |  |
| --- | --- |
| 제 목 | 시계 구현 + 코사인 법칙 구현 |
| 과 목 | 컴퓨터 개론 및 실습 |
| 날짜(년/월/일) | 2017/04/06 |
| 소속 학과 | 컴퓨터 전자 시스템 공학부 |
| 학번 | 201702234 |
| 이름 | 유동혁 |

|  |  |
| --- | --- |
| 본 보고서의 내용 중 다른 문서(자료)를 인용한 것이 있습니까?  예 ( ) 아니오 ( V ) | |
| 위에서 ‘예’로 답한 경우, 인용한 다른 문서는 무엇인지 아래에 명시해 주세요.(여러 개의 경우 주요 자료 2개 까지) | |
| - 저자 1 : |  |
| * 제목 1 : |  |
| * 저자 2 : |  |
| * 제목 2 : |  |

1. **디지털 시계(24시간 기준)의 방식으로 경과시간후의 시각을 표시하는 프로그램**

**1.1. Solution**

- 시, 분, 초를 구성하는 숫자들은 정수이므로 int변수형, %d를 사용한다.

- 현재시각과 경과시간을 초로 변환해 더한 후 계산한다.

- 전체 초 = (전체 초 / 60 / 60 % 24)시간 (전체 초 / 60 % 60)분 (전체 초 % 60)초

-

**1.2 Source code**

/\*

\* sigye.c

\*

\* Created on: 2017. 4. 1.

\* Author: 유동혁

\*/

#include<stdio.h>

#define hpd 24//hour per day

#define mph 60//minute per hour

#define spm 60//second per minute

int main()//시, 분, 초를 입력 받고 다음에 입력 받는 시, 분, 초가 지난 시간을 출력하는 프로그램

{

setvbuf(stdout, NULL, \_IONBF, 0);

int c\_hour, c\_min, c\_sec, ct\_sec;

int l\_hour, l\_min, l\_sec, lt\_sec;

int r\_hour, r\_min, r\_sec, rt\_sec;

printf("Enter current hour(0~23) : ");

scanf("%d", &c\_hour);

printf("Enter current minute(0~59) : ");

scanf("%d", &c\_min);

printf("Enter current second(0~59) : ");

scanf("%d", &c\_sec);

ct\_sec = (3600 \* c\_hour) + (60 \* c\_min) + c\_sec;//현재 시각을 초로 변환

printf("Enter lapse hour(0~23) : ");

scanf("%d", &l\_hour);

printf("Enter lapse minute(0~59) : ");

scanf("%d", &l\_min);

printf("Enter lapse second(0~59) : ");

scanf("%d", &l\_sec);

lt\_sec = (3600 \* l\_hour) + (60 \* l\_min) + l\_sec;// 경과 시간을 초로 변환

rt\_sec = ct\_sec + lt\_sec;//현재 시간, 경과 시간을 더한 초

r\_sec = rt\_sec % spm;

r\_min = (rt\_sec / spm) % mph;

r\_hour = (rt\_sec / spm)/ mph % hpd;

//전체 초 = (전체 초 / spm / mph % hpd)시간 (전체 초 / spm % mph)분 (전체 초 % spm)초

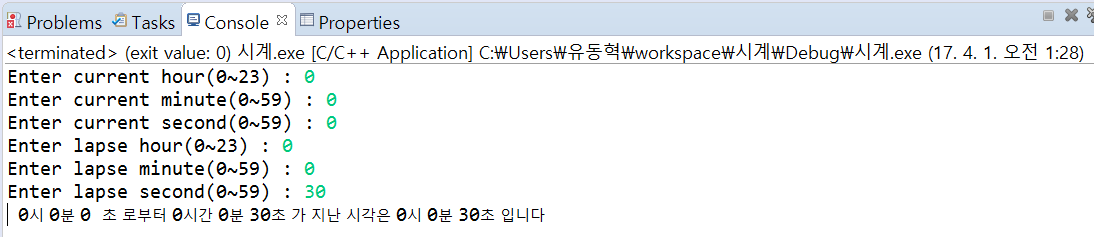
printf(" %d시 %d분 %d 초 로부터 %d시간 %d분 %d초 가 지난 시각은 %d시 %d분 %d초 입니다", c\_hour, c\_min, c\_sec, l\_hour, l\_min, l\_sec, r\_hour, r\_min, r\_sec);

return 0;

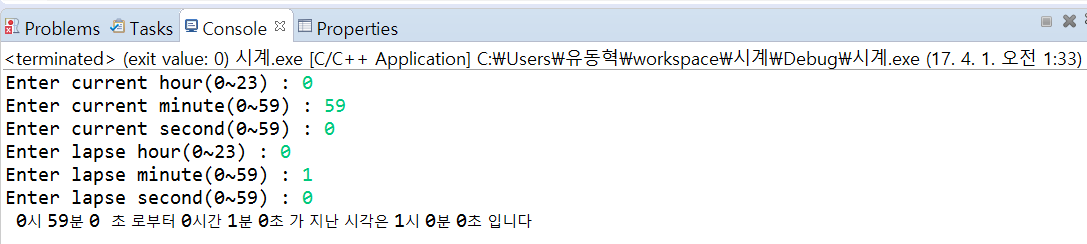
}

**1.3 Result(Snapshot)**

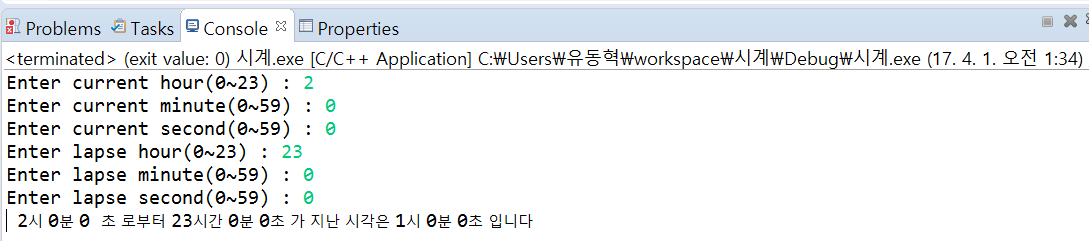
↓현재시각(00:00:00) 경과시간(00:00:30)



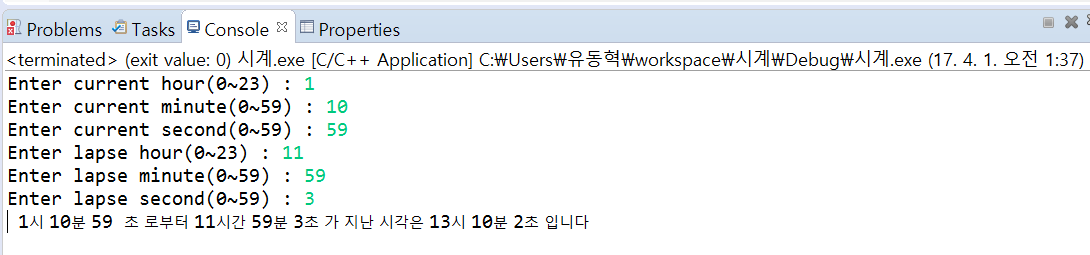
↓현재시각(00:59:00) 경과시간(00:01:00)



↓현재시각(02:00:00) 경과시간(23:00:00)



↓현재시각(01:10:59) 경과시간(11:59:03)



**1.4 Analysis & Discussion**

- 현재시간, 경과시간을 초로 변환하지않고 계산하는 방법은?

- 사용한 변수가 총 12가지인데(c\_hour, c\_min, …) 그 수를 줄일 수 있을까?

-

**2. 코사인 법칙을 이용하여 삼각형의 세 변의 길이를 입력 받으면 세 각의 크기를 알려주는 프로그램**

**2.1 Solution**

- acos함수를 이용하기 위해 헤더파일<math.h>를 선언한다.

- 코사인 법칙 : a^2 = b^2 + c^2 - 2bc\*cosA

- acos함수는 결과값이 rad이기 때문에 180/PI를 곱해 degree로 변환한다.

- ang\_A,B,C의 값을 소수점 첫째자리 까지만 출력하기위해 %.1f를 사용한다.

**2.2 Source code**

/\*

\* cos rule.c

\*

\* Created on: 2017. 4. 1.

\* Author: 유동혁

\*/

#include<stdio.h>

#include<math.h>//acos함수를 이용하기 위함

#define PI 3.141592

int main()//코사인 법칙에 따라, 삼각형의 세 변의 길이를 입력 받으면 세 각의 크기(degree)를 알려주는 프로그램

{

setvbuf(stdout, NULL, \_IONBF, 0);

float AB, BC, CA;

float ang\_A, ang\_B, ang\_C;

printf("Enter AB length of tri.ABC : ");

scanf("%f", &AB);

printf("Enter BC length of tri.ABC : ");

scanf("%f", &BC);

printf("Enter CA length of tri.ABC : ");

scanf("%f", &CA);

ang\_A = (180 / PI) \* acos((-(BC \* BC) + (CA \* CA) + (AB \* AB)) / (2 \* CA \* AB));

ang\_B = (180 / PI) \* acos((-(CA \* CA) + (AB \* AB) + (BC \* BC)) / (2 \* AB \* BC));

ang\_C = (180 / PI) \* acos((-(AB \* AB) + (BC \* BC) + (CA \* CA)) / (2 \* BC \* CA));

/\*

\* a^2 = b^2 + c^2 - 2bc\*cosA

\* 🡺 cosA = (-a^2 + b^2 + c^2) / 2bc

\* 🡺 acos{(-a^2 + b^2 + c^2) / 2bc} = A

\*/

//ang\_A,B,C는 acos만 사용했을 때 rad값으로 나오기 때문에 180 / PI을 곱해 degree로 변환한다

printf("\n");

printf("Angle A degree : %.1f\n", ang\_A);

printf("Angle B degree : %.1f\n", ang\_B);

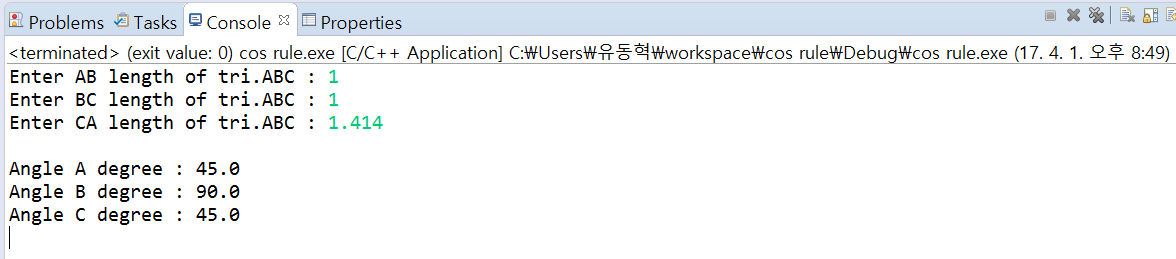
printf("Angle C degree : %.1f\n", ang\_C);

return 0;

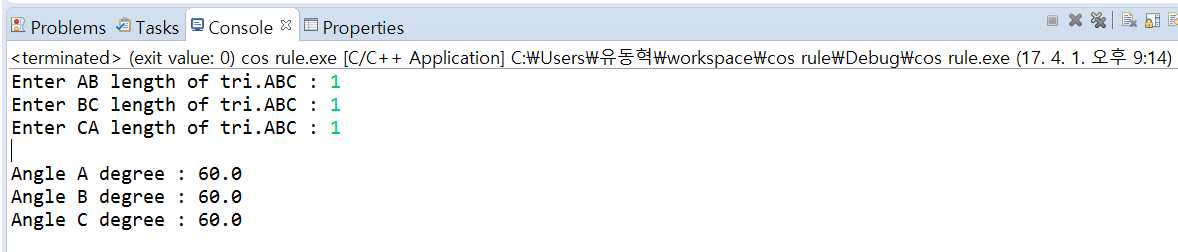
}

**2.3 Result(Snapshot)**

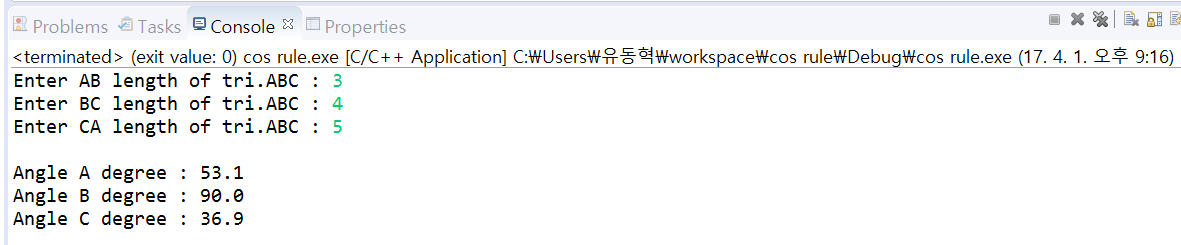
↓ AB : 1, BC : 1, CA : 1.414



↓ AB : 1, BC : 1, CA : 1



↓ AB : 3, BC : 4, CA : 5



**2.4 Analysis & Discussion**

- 거듭제곱을 할 때 a\*a가 아닌 a^2를 사용하는 방법은?

-

-

-