# 창의적 소프트웨어 설계 실습 문제 hw11 – hw11-2

**제출 기한**

12월 3일 (월) 11:59PM

1. hw11-2(mkdir hw11-2)라는 폴더를 만들고 GitLab에 push
2. hw11-2 디렉토리에 Makefile과 각 문제에서 요구하는 파일들을 작성
3. **‘make’명령을 수행하여 숙제가 모두 빌드**
4. 최종 버전을 GitLab에 commit
5. **시간과 파일명, 입력과 출력 방식 반드시 지키기**

## 과제1. 시계 (WorldClock)

날짜는 저장하지 않고, 24시간을 표시하는 시계를 작성한다. 시계는 SetTime(h, m, s)으로 시간 설정하거나 (h = 시간, m = 분, s = 초) Tick(s) 함수를 이용하여 현재 시간에 s만큼 시간을 추가할 수 있다. 주어진 선언을 기반으로 작성한다.

class WorldClock {

public:

void Tick(int seconds = 1);

bool SetTime(int hour, int minute, int second);

// 잘못된 값 입력시 false 리턴하고 원래 시간은 바뀌지 않음.

int hour() const;

int minute() const;

int second() const;

private:

// 필요시 변수 추가

};

// hh:mm:ss 형식으로 입출력

ostream& operator<<(ostream& os, const WorldClock& c);

// hh:mm:ss 로 입력받음.

istream& operator>>(istream& is, WorldClock& c);

\* 필요시 private 멤버 변수 및 함수 추가 가능. friend는 사용금지.

설계 시 고려사항

* 멤버 변수는 일반적으로 두가지 방법을 고려할 수 있음. 각 방법의 장단점을 고려해 결정.
  + 시, 분, 초 정보를 따로 저장하는 방법.
  + 초 정보만 저장하고 시, 분을 계산해 내는 방법 (예: 10000초 = 2 시간 46 분 40 초).
* set hh:mm:ss : 현재 timezone에서 hh시 mm분 ss초로 설정하고 출력.
  + 잘못된 입력이 들어오면 "Invalid time: <hh:mm:ss>" 를 출력.
* tick sec: 현재 timezone에 주어진 초만큼 더하고 출력.
  + 초는 정수형으로 들어옴 (음수 가능)

파일명 : world\_clock (world\_clock.h, world\_clock.cc, world\_clock\_main.cc)

입력 / 출력

|  |
| --- |
| **$ ./world\_clock**  **set 1:20:00**  **1:20:0**  **tick 13600**  **5:6:40**  **tick 10000**  **7:53:20**  **tick -10000**  **5:6:40**  **tick -30000**  **20:46:40**  **tick 40000**  **7:53:20**  **set 123:34:56**  **Invalid time: 123:34:56**  **set 24:60:60**  **Invalid time: 24:60:60**  **set 0:0:0**  **0:0:0**  **quit** |

## 과제2. 달력 (Calendar)

연, 월, 일 정보를 저장하는 달력 클래스를 작성한다. 윤년도 처리할 수 있어야 한다. 달력은 SetDate(year, month, day)로 날짜를 설정하거나 NextDay(n) 함수를 이용하여 현재 날짜에 n일 만큼을 추가할 수 있다. 주어진 선언을 기반으로 작성한다.

class Date {

public:

Date();

Date(int year, int month, int day);

void NextDay(int n = 1);

bool SetDate(int year, int month, int day);

int year() const;

int month() const;

int day() const;

private:

// 윤년을 판단하여 주어진 연도에 해당하는 날짜 수(365 또는 366)를 리턴.

// 2월을 29일로 둔 해를 윤년이라고 함

static int GetDaysInYear(int year);

// 해당 날짜가 해당 연도의 처음(1월 1일)부터 며칠째인지를 계산.

static int ComputeDaysFromYearStart(int year, int month, int day);

int year\_, month\_, day\_;

};

struct InvalidDateException {

string input\_date;

InvalidDateException(const string& str) : input\_date(str) {}

};

// yyyy.mm.dd 형식으로 입출력.

// 사용자 입력 오류시 >> operator는 InvalidDateException을 발생할 수 있음.

ostream& operator<<(ostream& os, const Date& c);

istream& operator>>(istream& is, Date& c);

\* 필요시 private 멤버 변수 및 함수에 한해서 변경, 추가 가능.

설계 시 고려사항

* 달력 클래스 구현 - 연,월,일 정보 저장. (윤년 처리)

윤년 판단 조건

* 4로 나누어 떨어지는 경우는 윤년이다.
* 단 100으로 나누어 떨어지는 경우는 윤년이 아니다. (400으로 나누어 떨어지는 경우를 제외)
* 100으로 나누어 떨어지는 값 중에 400으로 나누어 떨어지는 연도도 윤년이다.
* SetDate(y, m, d)로 날짜 설정.
* NextDay(n)으로 날짜 증가,감소 (임의의 양수, 음수 값이 주어질 수 있음).
* NextDay() 의 경우 다음과 같은 알고리즘 사용을 고려.
  + 현재 날짜가 지금 연도에서 며칠째인지를 계산하고 주어진 n 을 더하여 days 값을 구함.
  + days 값이 음수인 경우 연도를 감소하고 해당년도의 날짜수를 days에 더함.
  + days 값이 해당년도의 날짜수보다 클 경우 해당년도의 날짜수를 빼고 연도를 증가.
  + 연도를 찾으면 해당년도에서 1월부터 그 달의 날짜수를 빼면서 월 을 증가.
  + 나머지는 일 에 더하여 완성.
* set yyyy.mm.dd : yyyy 년 mm 월 dd 일로 현재 날짜를 설정하고 출력.
* 잘못된 입력이 들어오면 "Invalid date: <yyyy.mm.dd>" 를 출력.
* next day 또는 next <offset> : 다음날 또는 offset만큼으로 날짜 변경.
* quit : 프로그램 종료.

파일명 : calendar (calendar.h, calendar.cc, calendar\_main.cc)

입력 / 출력

|  |
| --- |
| **$ ./calendar**  **set 2013.1.31**  **2013.1.31**  **next\_day**  **2013.2.1**  **next\_day**  **2013.2.2**  **set 2011.12.29**  **2011.12.29**  **next\_day**  **2011.12.30**  **next\_day**  **2011.12.31**  **next\_day**  **2012.1.1**  **set 2013.2.29**  **Invalid date: 2013.2.29**  **next\_day**  **2012.1.2**  **set 2012.2.29**  **2012.2.29**  **next\_day**  **2012.3.1**  **quit**  **$**  **$ ./calendar**  **set 0.0.0**  **Invalid date: 0.0.0**  **set 0.1.1**  **0.1.1**  **next -1**  **-1.12.31**  **next 3**  **0.1.3**  **quit**  **$** |