2023 Bahar Dönemi BİL142 Proje-1

Proje hazırlanırken Necati Ergin Hoca'nın (https://github.com/necatiergin/cpp_odev/tree/main/040_CLASS_DATE) belgelerinden faydalanılarak hazırlanmıştır.

 # Date sınıfı

Aşağıda ismi `Date` olan bir sınıfın tanımlandığı başlık dosyası yer almaktadır.
Bu projede `Date` sınıfının kodlarını yazmanız isteniyor.
`Date` sınıfı türünden bir nesnenin değeri bir tarihtir. Örnek: `15 Şubat 1998`

Aşağıdaki açıklamalar kodda bulunan yorum satırlarına ilişkindir: (Açıklamaların sonunda bir taslak kod bulunmaktadır.)

1. Sınıfın hizmet verdiği en küçük yıl değeri
2. `random_date` işlevinin üreteceği tarih için en küçük yıl değeri
3. `random_date` işlevinin üreteceği tarih için en büyük yıl değeri
4. Haftanın günü için `enum class` türü
5. Varsayılan kurucu işlev: `Date` nesnesini `01-01-1900` tarihi ile oluşturacak
6. `Date` nesnesini gün, ay, yıl değeri ile oluşturacak kurucu işlev
7. `Date` nesnesini formatlanmış yazıdan alacağı tarih değeri ile oluşturacak. Format: `gg/aa/yıl`
8. `Date` nesnesini "calender time" değerinden dönüştüreceği tarih değeri ile oluşturacak.
9. Ayın gününü döndürüyor.
10. Ay değerini döndürüyor. (Ocak 1, Şubat 2, ...)
11. Tarihin yıl değerini döndürüyor
12. Yılın gününü döndürüyor (01 Ocak ---> 1 31 Aralık---> 365 ya da 366
13. Haftanın gününü döndürüyor.
14. Tarihin ayın günü değerini değiştiriyor.
15. Tarihin ayını değiştiriyor
16. Tarihin yılını değiştiriyor.
17. Tarihi değiştiriyor.
18. Tarihten gün çıkartan `const` üye operatör işlevi. Geri dönüş değeri elde edilen tarih olacak.
19. Tarihi gelen gün kadar arttıran üye operatör işlevi. Geri dönüş değeri `*this` olmalı.
20. Tarihi gelen gün kadar eksilten üye operatör işlevi. Geri dönüş değeri `*this` olmalı.
21. Önek `++` operatörünü yükleyen işlev. (İşlevin referans döndürdüğüne dikkat ediniz).
22. Sonek `++` operatörünü yükleyen işlev. (İşlevin referans döndürmediğine dikkat ediniz).
23. Önek `--` operatörünü yükleyen işlev. (İşlevin referans döndürdüğüne dikkat ediniz).
24. Sonek `--` operatörünü yükleyen işlev. (İşlevin referans döndürmediğine dikkat ediniz).
25. Rastgele bir tarih döndüren sınıfın `static` üye işlevi.
26. Artık yıl testi yapan sınıfın `static` üye işlevi.
27. `Date` nesnelerinin karşılaştırılmasını sağlayacak global operatör işlevleri
28. İki tarih arasındaki gün farkını döndüren global operatör işlevi
29. Gelen tarihten `n` gün sonrasını döndüren global operatör işlevleri
30. İçsel `(nested) enum class Weekday` için arttırma ve eksiltme işlevleri
31. Date nesnelerinin değerlerini çıkış akımlarına yazdıracak global operatör işlevi `(inserter)` Formatlama şöyle olmalı: `31 Ekim 2019 Persembe`

32. `Date` nesnelerine giriş akımlarından aldığı karakterlerden oluşturulacak değeri yerleştiren global operatör işlevi `(extractor)`
Formatlama: `gg/aa/yyyy` (ayıraç olarak istenilen bir karakter kullanılabilir.

Diğer notlar:
* Dilediğiniz global işlevleri `"friend"` yapabilirsiniz.
* Sınıfın `private` arayüzünü dilediğiniz gibi oluşturabilirsiniz. Yalnız veri elemanlarını (m_year, m_day, m_month) değiştirmeyiniz.
* Gerekli görürseniz sınıfın `public` arayüzüne eklemeler yapabilirsiniz.
* Dilediğiniz işlevleri `"constexpr"` yapabilirsiniz.
* Bu ödevde `"exception handling"` araçlarını kullanmayacağız. (`exception handling` konusunu gördükten sonra kodlarda değişiklik yapacağız.) Fonksiyonlara gönderilen argümanların uygun ve doğru değerler olduğunu varsayabilirsiniz.
* Kod tekrarından mümkün olduğu kadar kaçınmalısınız.
* `const` doğruluğuna `(const correctness)` çok dikkat etmelisiniz. `(const olması gereken tüm varlıklar const olmalı)`
* Gerekli olduğunu düşündüğünüz yerlerde [[nodiscard]] attribute'unu kullanın.

TASLAK KOD

```
#ifndef DATE_H
#define DATE_H
#include <iosfwd>
#include <ctime>

namespace project {
class Date {
public:
	static constexpr int year_base = 1900;  //1
	static constexpr int random_min_year = 1940;  //2
	static constexpr int random_max_year = 2020;  //3
	enum class Weekday {Sunday, Monday, Tuesday, Wednesday, Thursday, Friday, Saturday}; //4
	Date(); //5
	Date(int d, int m, int y);  //6
	Date(const char *p);  //7
	Date(std::time_t timer); //8
	int get_month_day()const; //9
	int get_month()const; //10
	int get_year()const; //11
	int get_year_day()const; //12
	Weekday get_week_day()const; //13

	Date& set_month_day(int day); //14
	Date& set_month(int month); //15
	Date& set_year(int year); //16
	Date& set(int day, int mon, int year); //17
	Date operator-(int day)const; //18
	Date& operator+=(int day); //19
	Date& operator-=(int day); //20
```

```
        Date& operator++(); //21
        Date operator++(int); //22
        Date& operator--(); //23
        Date operator--(int); //24

        friend bool operator<(const Date &, const Date &); //27
        friend bool operator==(const Date &, const Date &); //27
        static Date random_date(); //25
        static constexpr bool isleap(int y); //26

        friend std::ostream &operator<<(std::ostream &os, const Date &date); //31
        friend std::istream &operator>>(std::istream &is, Date &date); //32
};

bool operator<(const Date &, const Date &); //27
bool operator<=(const Date &, const Date &); //27
bool operator>(const Date &, const Date &); //27
bool operator>=(const Date &, const Date &); //27
bool operator==(const Date &, const Date &); //27
bool operator!=(const Date &, const Date &); //27

int operator-(const Date &d1, const Date &d2); //28
Date operator+(const Date &date, int n); //29
Date operator+(int n, const Date &); //29
Date::Weekday& operator++(Date::Weekday &r); //30
Date::Weekday operator++(Date::Weekday &r, int); //30
Date::Weekday& operator--(Date::Weekday &r); //30
Date::Weekday operator--(Date::Weekday &r, int); //30}

#endif
```