



Sign up



Code

Issues 0

Pull requests 0

Projects 0

Actions

Wiki

Security

Pulse

Branch: master

Find file

Copy path

cpp_kursu_odevleri / classes / class_12.md

Fetching contributors...



107 lines (94 sloc) | 5.5 KB

Raw

Blame

History



Aşağıda ismi *Date* olan bir sınıfın tanımlandığı başlık dosyası yer almaktadır. Bu ödevde *Date* sınıfının kodlarını yazmanız isteniyor:

Date sınıfı türünden bir nesnenin değeri bir tarihtir. Örnek: 15 Şubat 1998

Aşağıdaki açıklamalar kodda bulunan yorum satırlarına ilişkindir:

- //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 yazından alacağı tarih değeri ile oluşturacak. Format: *gg/aa/yıl*
- //8 *Date* nesnesini "*calender time*" değerinden dönüştürecek 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 gününü değiştiriyor.
- //15 Tarihin ayını değiştiriyor.
- //16 Tarihin yılını değiştiriyor.
- //17 Tarihten gün çıkartan *const* üye operatör işlevi. Geri dönüş değeri elde edilen tarih olacak.*
- //18 Tarihi gelen gün kadar arttıran üye operatör işlevi. Geri dönüş değeri **this* olmalı.*
- //19 Tarihi gelen gün kadar eksiltten üye operatör işlevi. Geri dönüş değeri **this* olmalı.*
- //20 Önek ++ operatörünü yükleyen işlev. (*İşlevin referans döndürdüğüne dikkat ediniz*).
- //21 Sonek ++ operatörünü yükleyen işlev. (*İşlevin referans döndürmediğine dikkat ediniz*).
- //22 Önek -- operatörünü yükleyen işlev. (*İşlevin referans döndürdüğüne dikkat ediniz*).
- //23 Sonek -- operatörünü yükleyen işlev. (*İşlevin referans döndürmediğine dikkat ediniz*).
- //24 Rastgele bir tarih döndüren sınıfın *static* üye işlevi.

- //25 Artık yıl testi yapan sınıfın *static* üye işlevi.
- //26-31 *Date* nesnelerinin karşılaştırılmasını sağlayacak global operatör işlevleri
- //32 İki tarih arasındaki gün farkını döndüren global operatör işlevi
- //33-34 Gelen tarihten *n* gün sonrasını döndüren global operatör işlevleri
- //35-38 İçsel (*nested*) *enum class Weekday* için arttırma ve eksiltme işlevleri
- //39 *Date* nesnelerinin değerlerini çıkış akımlarına yazdıracak global operatör işlevi (*inserter*) Formatlama şöyle olmalı: 31 Ekim 2019 Persembe
- //40 *Date* nesnelere 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.
- Gerekli görürseniz sınıfın *public* arayüzüne eklemeler yapabilirsiniz.
- Gerekli görürseniz sınıfın *public* arayüzünde değişiklikler yapabilirsiniz.
- Sınıfın *public* öğelerinin isimlerini istediğiniz şekilde değiştirebilirsiniz.
- Dilediğiniz işlevleri "*constexpr*" yapabilirsiniz.
- Bu ödevde "*exception handling*" araçlarını kullanmayacağız. (*exception handling* konusunu gördükten sonra kodlarda değişiklik yapacağız.)
- Kod tekrarından mümkün olduğu kadar kaçınmalısınız.
- *const* doğruluğuna (*const correctness*) çok dikkat etmelisiniz. (*const* olması gereken tüm varlıklar *const* olmalı)
- Gereksiz yorum satırlarından mümkün olduğu kadar kaçınmalısınız.

```

#include <iosfwd>
#include <ctime>

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 operator-(int day)const; //17
    Date& operator+=(int day); //18
    Date& operator-=(int day); //19
    Date& operator++(); //20
    Date operator++(int); //21
    Date& operator--(); //22
    Date operator--(int); //23

    static Date random_date(); //24
    static constexpr bool isleap(int y); //25
};

bool operator<(const Date &, const Date &); //26
bool operator<=(const Date &, const Date &); //27
bool operator>(const Date &, const Date &); //28
bool operator>=(const Date &, const Date &); //29
bool operator==(const Date &, const Date &); //30
bool operator!=(const Date &, const Date &); //31

int operator-(const Date &d1, const Date &d2); //32
Date operator+(const Date &date, int n); //33
Date operator+(int n, const Date &); //34
Date::WeekDay& operator++(Date::WeekDay &r); //35
Date::WeekDay& operator++(Date::WeekDay &r, int); //36
Date::WeekDay& operator--(Date::WeekDay &r); //37
Date::WeekDay& operator--(Date::WeekDay &r, int); //38

std::ostream &operator<<(std::ostream &os, const Date &date); //39
std::istream &operator>>(std::istream &is, Date &date); //40

```

[Contact GitHub](#)
[Pricing](#)
[API](#)
[Training](#)
[Blog](#)
[About](#)