

C++ - Modül 05

Tekrar ve İstisnalar

Özet: Bu belge, C++ modüllerinden Modül 05'in alıştırma ve örneklerini içerir.

Sürüm: 9.1

İçindekiler

BEN	giriş	2
III	Genel kurallar	3
III	Egzersiz 00: Anne, büyüdüğümde bürokrat olmak istiyorum! 5	
IV	Alıştırmalar 01: Toplanı n kurtçuklar!	7
V	Egzersiz 02: Hayır, 28B formuna ihtiyacı nı z var, 28C'ye değ il...	9
VI	Alıştırmalar 03: En azı ndan bu, kahve yapmayı yener	11

Bölüm I

giriş

C++, Bjarne Stroustrup tarafı ndan C programlama dilinin veya "C with Classes"ı n (kaynak: [Wikipedia](#)) bir uzantı sı olarak oluş turulan genel amaçlı bir programlama dilidir .

Bu modüllerin amacı sizi Nesne Yönelimli Programlama ile tanı ş tı rmaktır .
Bu, C++ yolculuğ unuzun baş langı ç noktası olacaktır . OOP öğ renmek için birçok dil önerilir.
Eski dostunuz C'den türetildiğ i için C++'ı seçmeye karar verdik.
Bu karmaş ık bir dil olduğ undan ve iş leri basit tutmak için kodunuz C++98 standardı na uygun olacaktır .

Modern C++'ı n pek çok ağı dan çok farklı olduğ unun farkı ndayız . Dolayısı yla, yetkin bir C++ geliř tiricisi olmak istiyorsanız, 42 Ortak Çekirdekten sonra daha da ileri gitmek size kalmı ş !

Bölüm II

Genel kurallar

derleme

- Kodunuzu c++ ve -Wall -Wextra -Werror iş aretleri ile derleyin
- -std=c++98 bayrağı nı eklerseniz kodunuz yine de derlenmelidir.

Biçimlendirme ve adlandırma kuralları

- Alıştırmaları dizinleri şu şekilde adlandırılacaktır: ex00, ex01, ... , eski
- Dosyaları nı zı , sı nı fları nı zı , iş levlerinizi, üye iş levlerinizi ve özniteliklerinizi gerektiği gibi adlandırın. yardımcı notlar.
- Sınıf isimlerini UpperCamelCase formatında yazın. Sınıf kodu içeren dosyalar her zaman sınıf adı na göre adlandırılır. Örneğin: ClassName.hpp/ClassName.h, ClassName.cpp veya ClassName.hpp. Ardından, bir tuğ la duvarı temsil eden "BrickWall" sınıfı nı tanı mını içeren bir baş lı k dosyanı z varsa, adı BrickWall.hpp olacaktır.
- Aksi belirtilmedikçe, her gı ş mesajı yeni bir satı rla sonlandırılmalıdır. karakter ve standart gı ktı ya görüntülenir.
- Elveda Norminette! C++ modüllerinde kodlama stili uygulanmaz. En sevdiğ inizi takip edebilirsiniz. Ancak, akran değ erlendiricilerinizin anlayamadı ğ ı bir kodun not veremediğ i bir kod olduğ unu unutmayı n. Temiz ve okunabilir bir kod yazmak için elinizden geleni yapın.

İ zin Verildi/Yasaklandı

Artık C'de kodlama yapmı yorsunuz. C++ zamanı ! Öyleyse:

- Standart kitaplı ktan hemen hemen her ş eyi kullanmanı za izin verilir. Bu nedenle, zaten bildiklerinize bağı lı kalmak yerine, alış kı n olduğ unuz C iş levlerinin C++-ish sürümlerini mümkün olduğ unca kullanmak akı llı ca olacaktır.
- Ancak, baş ka bir harici kitaplı k kullanamazsınız. Bu, C++11 (ve türetilmiş formlar) ve Boost kitaplı kları nı n yasak olduğ u anlamı na gelir. Aş ağı daki iş levler de yasaktı r: *printf(), *alloc() ve free(). Bunları kullanı rsanız notunuz 0 olur ve o kadar.

- Aksi açıkça belirtilmediği sürece <ns_name> ve arkadaş anahtar kelimeleri yasaktır. Aksi takdirde notunuz -42 olacaktır.
- STL'yi yalnızca Modül 08 ve 09'da kullanmanıza izin verilir. Bunun anlamı: o zamana kadar Konteyner (vektör/liste/harita/vb.) ve Algoritma (<algorithm> başlılığı) içermesini gerektiren herhangi bir şey olmaması. Aksi takdirde notunuz -42 olacaktır.

Birkaç tasarım gereksinimi

- Bellek sıralanması C++'da da meydana gelir. Bellek ayrıldığı anda (yeni anahtar kelime), bellek sıralamalarından kaçınılmalıdır.
- Modül 02'den Modül 09'a kadar sıranın fları nız Ortodoks olarak tasarlanmalıdır. Kanonik Form, aksi açıkça belirtilmedikçe.
- Bir başlık dosyasına konulan herhangi bir işlev uygulaması (işlev şablonları hariç), alıştırma için 0 anlamına gelir.
- Başlıkları nızın her birini diğerlerinden bağımsız olarak kullanabilmelisiniz. Bu nedenle, ihtiyaç duydukları tüm bağımlılıkları içermeleri gerekir. Ancak, içme korumaları ekleyerek çift içme probleminden kaçınılmalıdır. Aksi takdirde notunuz 0 olacaktır.

beni oku

- Gerekirse (örn. kodunuzu bölmek için) bazı ek dosyalar ekleyebilirsiniz. Bu ödevler bir program tarafından doğrulanmadığından, zorunlu dosyaları teslim ettiğiniz sürece bunu yapmaktan çekinmeyin.
- Bazen, bir alıştırmanın yönergeleri kısa görünebilir, ancak örnekler şunları gösterebilir: talimatlarda açıkça yazılmayan gereksinimler.
- Başlamadan önce her modülü baştan sona okuyun! Gerçekten, yap.
- Odin adını, Thor adını! Beynini kullan!!!




Çok sayıda sıranı uygulamanız gerekecek. Bu sıkıcı görünebilir, favori metin düzenleyicinizi yazamazsanız.



Alıştırmaları tamamlamanız için size belirli bir miktar özgürlük verilir. Ancak zorunlu kurallara uyun ve tembel olmayın. Yapabilirdin birçok yararlı bilgiyi kaçırmıyorsun! Hakkında okumaktan çekinmeyin teorik kavramlar.

Bölüm III

Egzersiz 00: Anne, büyüdüğ ümde bürokrat olmak istiyorum!

	Egzersiz : 00
Anne ben büyüünce bürokrat olmak istiyorum!	
Teslim dizini : ex00/ Teslim	
edilecek dosyalar : Makefile, main.cpp, Bureaucrat.{h, hpp}, Bureaucrat.cpp Yasak iş levler : Yok	



İ stisna sı nı fları nı n Ortodoks Kanonik Biçimde tasarlanması gerekmediğ ini lütfen unutmayı n. Ama diğ er tüm sı nı flar yapmak zorunda.

Ofisler, koridorlar, formlar ve bekleme kuyrukları ndan oluş an yapay bir kabus tasarlayalı m.
Kulağ a eğ lenceli mi geliyor? HAYIR? Çok kötü.

İ lk olarak, bu geniş bürokratik makinedeki en küçük diş li ile baş layı n: Bürokrat .

Bir Bürokrat ş unlara sahip olmalı dı r:

- Sabit bir ad.
- Ve 1 (mümkün olan en yüksek not) ile 150 (mümkün olan en düş ük not) arası nda değ iş en bir not seviye).

Geçersiz bir derece kullanarak bir Bürokrat örneğ ini oluş turmaya yönelik tüm giriş imler bir istisna oluş turmalı dı r: ya bir Bureaucrat::GradeTooHighException ya da bir Bureaucrat::GradeTooLowException.

Bu özneliklerin her ikisi için de alıcılar sağlayacaktır: getName() ve getGrade(). Bürokrat notunu artırmak veya azaltmak için iki üye işlevi de uygulayın. Derece aralığı dışında ise, her ikisi de oluşturucuyla aynı istisnaları atar.



Hatırlamak. 1. derece en yüksek, 150 en düşük olduğu için, 3. notu artırmak bürokrata 2. notu vermelidir.

Fırlatılan istisnalar, try ve catch blokları kullanılarak yakalanabilir olmalıdır:

```
dene
{
    /* bürokratlarla bir şeyler yapalım */

} catch (std::exception &e) {

    /* istisnayı ele al */
}
```


Gibi bir şey yazdırmak için ekleme («) operatörünün aşırı yüklenmesini uygulayacaktır. (ağır ayraç olmadan):

<isim>, bürokrat sınıfını fi <derece>.

Her zamanki gibi, her şeyin beklendiği gibi çalışmasını kanıtlamak için bazı testler yapılır.

Bölüm IV

Alıştırmalar 01: Toplanı n kurtçuklar!

	Egzersiz : 01
Toplanı n kurtçuklar!	
Teslim dizini : ex01/ Teslim	
edilecek dosyalar : Önceki alıştırmadan dosyalar + Form.{h, cpp}, Form.cpp Yasak işlevler : Yok	

Artık bürokratları nı z oldu ğ una göre, onlara yapacak bir şeyler verelim. Ne daha iyi aktivite bir yıl ı n form doldurmaktan daha fazlası olabilir mi?

Ardı ndan bir Form sınıfı oluşturulmuş. Ş unlara sahiptir:

- Sabit bir ad.
- İ mzanın p imzalanmadı ğ ı nı gösteren bir boole (yapı m aş aması nda değ il).
- İ mzalamak için gereken sabit bir not.
- Ve bunu gerçekleştirmek için gereken sabit not.

Tüm bu özellikler özeldir, korunmaz.

Formun notları , Bürokrat için geçerli olan aynı kuralları izler. Böylece, bir form notu sınıfı nı rları n dı ş ı ndaysa aş ağı daki istisnalar atılır: Form::GradeTooHighException ve Form::GradeTooLowException.

Daha önce oldu ğ u gibi, tüm öznitelikler için alıcı lar yazı n ve tüm formun bilgilerini basan ekleme («) operatörünün aş ı rı yüklenmesi.

Parametre olarak Bürokrat alan Forma ayrıca bir beSigned() üye işlevi ekleyin. Bürokratın notu yeterince yüksekse (gerekli olandan daha yüksek veya eşit) form durumunu imzalı olarak değiştirir. Unutmayın, 1. derece 2. dereceden daha yüksektir. Not çok düşüğe bir Form::GradeTooLowException atılır.

Son olarak, Bureaucrat'a bir signForm() üye işlevi ekleyin. Form imzalandıysa, şöyle bir şey yazdıracak:

<bürokrat> imzalı <form>


Aksi takdirde, şöyle bir şey yazdıracaktır:

<bürokrat>, <sebep> nedeniyle <form>'u imzalayamadı.

Her şeyin beklendiği gibi çalıştığından emin olmak için bazı testler uygulayın ve teslim edin.

Bölüm V

Egzersiz 02: Hayır, 28B formuna ihtiyacı nı z var, 28C'ye değ il...

	Egzersiz : 02
Hayır, 28B formuna ihtiyacı nı z var, 28C'ye değ il...	
Teslim dizini : ex02/ Teslim	
edilecek dosyalar : Makefile, main.cpp, Bureaucrat.{h, hpp}, Bureaucrat.cpp + AForm.{h, hpp}, ShrubberyCreationForm.{h, hpp}, + RobotomyRequestForm.{h, hpp}, PresidentialPardonForm.{h, hpp}	
Yasaklanan iş levler: Yok	

Artı k temel formlara sahip olduđ unuza göre, gerçekten bir ş eyler yapan birkaç tane daha yapmanı n zamanı geldi.

Her durumda, Form temel sı nı fı soyut bir sı nı f olmalı dı r ve bu nedenle AForm olarak yeniden adlandırılmalı dı r. Form özneliklerinin gizli kalması gerektiđ ini ve temel sı nı f ta oldukları nı unutmayı n.

Aş ağı daki somut sı nı fları ekleyin:

- ShrubberyCreationForm: Gerekli notlar: imza 145, yürütme 137
Çalı ş ma dizininde bir <target>_shrubbery dosyası olu ş turun ve içine ASCII ağ ađ arı yazar.
- RobotomyRequestForm: Gerekli dereceler: iş aret 72, yürütme 45 Bazı delme sesleri ğ karı r. Ardı ndan, <target> öđ esinin zamanı n %50'sinde baş arı yla robotize edildiđ ini bildirir. Aksi halde, robotomy'nin baş arı sı z olduđ unu bildirir.
- PresidentialPardonForm: Gerekli notlar: imza 25, yönetici 5
<target> öđ esinin Zaphod Beeblebrox tarafı ndan affedildiđ ini bildirir.

Hepsi yapı cı ları nda yalnız ca bir parametre alı r: formun hedefi. İ ğ in örneđ in, evde çalı lı k dikmek istiyorsanı z "ev".

Şimdi, executive(Bureaucrat const & executor) const üye işlevini temel forma ekleyin ve formun somut sınıfı nı f eylemini yürütmek için bir işlev uygulayın. Formun imzalanıp imzalanmadığı nı ve formu uygulamaya çalışırken bürokratin notunun yeterince yüksek olup olmadığını kontrol etmelisiniz. Aksi takdirde, uygun bir istisna atın.

Her somut sınıf nı ftaki veya temel sınıf nı ftaki gereksinimleri kontrol etmek isteyip istemediğinizi (daha sonra formu yürütmek için başka bir işlevi çağırın) size kalmış. Ancak, bir yol olduğundan daha güzel.

Son olarak, executiveForm(Form const & form) üye işlevini Büroya ekleyin. Formu yürütmeye çalışmalıydır. Başarılı olursa, şöyle bir şey yazdırın:


```
<bürokrat> idam edildi <form>
```

Değilse, açık bir hata mesajı yazdırın.

Her şeyin beklendiği gibi çalıştığından emin olmak için bazı testler uygulayın ve teslim edin.

Bölüm VI

Alıştırmalar 03: En azından bu, kahve yapmayı yener

	Egzersiz : 03
En azından bu kahve yapmayı yener	
Teslim dizini : ex03/ Teslim	
edilecek dosyalar : Önceki alıştırmalardan dosyalar + Intern.{h, hpp}, Intern.cpp Yasak işlevler : Yok	

Form doldurmak yeterince cansız kıcı olduğ u için, bürokratları mızdan gün boyu bunu yapması nı istemek zalimlik olur. Neyse ki, stajyerler var. Bu alıştırmada, Intern sı nı fı nı uygulamalı sı nı z . Stajyerin adı , derecesi, benzersiz özellikleri yoktur. Bürokratları n umursadı ğ ı tek şey iş lerini yapmaları dır .

Ancak stajyerin önemli bir kapasitesi vardı r: makeForm() iş levi. İ ki dizi alır . Birincisi bir formun adı , ikincisi ise formun hedefidir. Hedefi ikinci parametreye baş latılacak olan bir Form nesnesine (adı parametre olarak iletilen olan) bir iş areti döndürür .

Ş unun gibi bir şey yazdı racak:

```
Stajyer <form> oluş turur
```

Parametre olarak iletilen form adı yoksa açık bir hata mesajı yazdı rır n.

if/elseif/else ormanı kullanmak gibi okunamayan ve çirkin çözümlerden kaçınmalıyız. Değerlendirme sürecinde bu tür şeyler kabul edilmeyecektir. Artık Piscine'de (havuzda) değilsiniz. Her zamanki gibi, her şeyin beklendiği gibi çalıştığını test etmeniz gerekir.

Örneğin, aşağıdaki kod "Ben der" üzerinde hedeflenen bir RobotomyRequestForm oluşturur:

```
{  
    Stajyer bazı RandomStajyer;  
    biçim* rf;  
  
    rrf = someRandomIntern.makeForm("robotomi talebi", "Bender");  
}
```