NYP

SOYUT SINIFLAR HAKKINDA BİLMEMİZ GEREKENLER

* Abstract methoda sahip olan bir sınıf abstract tanımlanmak zorundadır.
* Abstract class lar somut(concrete) methodlar da içerebilir.
* Soyut sınıflarda alan değişkenleri de olabilir(dobule c).
* Abstract class ların constructor u olabilir.
* Abstract class lardan nesne yaratılamaz.
* Eğer bir sınıfta constructor private ise o sınıftan nesne yaratılamaz.
* Soyut metot içeren sınıf, soyut olmak zorunda.
* Soyut metot sadece soyut sınıfta bulunabilir.
* Bir soyut sınıftaki soyut methot, yavru sınıflar tarafından gerçekleştirilmeli (overriding) aksi takdirde yavru sınıf da soyut tanımlanmalı.
* Soyut methotlar static tanımlanamaz!!!
* Bir soyut sınıftan nesne yaratılamaz.
* Soyut sınıflar soyut metot barındırmak zorunda değildir.
* Bir yavru sınıfta ata sınıf olan soyut sınıftan miras alınan concrete metot abstract bir metot ile ezilebilir (çok nadir) .
* Bir soyut sınıfın atası concrete (somut) sınıf olabilir.
* Soyut sınıflardan new ile nesne yaratamayız ama yine de onları veri tipi olarak kullanabiliriz.
* Soyut sınıflardan yeni nesneler yaratamadığımız halde konstrüktör içerebilirler .
* UML diyagramında methodun başında # işareti varsa protected demektir.
* UML diyagramında soyut sınıfların isimleri italik yazılır.
* UML diyagramında methodun altı çizili ise static olduğu anlamına gelir.
* OTO BOXİNG : 45 gibi bir int sayının otomatik olarak integer nesnesine dönüştürülmesine ya da 3445.53 gibi bir double sayının otomatik olarak double nesnesine dönüştürülmesine denir.

ARAYÜZLER HAKKINDA BİLMEMİZ GEREKENLER

* Arayüzler sadece sabitleri ve soyut metotları barındırır concrete (somut) metot barındıramazlar .
* Arayüzler de tıpkı soyut sınıflar gibi veri tipi olarak kullanılabilir.
* Arayüzlerden de tıpkı soyut sınıflarda olduğu gibi nesneler üretilemez .
* Bir sınıf sadece bir ata sınıftan (soyut/ somut) miras alabilir, ama pek çok arayüzü implement edebilir.
* bir arayüzden miras alınan tüm soyut metotlar ezilmelidir, aksi takdirde mirasçı sınıfın abstract tanımlanması gerekir.
* Interface lerdeki bütün methodlar **public abstract** tir. Eğer başına yazılmazsa java gizliden yazar.
* Interface lerdeki değişkenler **public static final** dır. Eğer başına yazılmazsa java gizliden yazar.
* Arayüzler başka arayüzleri extend edebilir ama normal sınıfları miras alamaz.
* Shellow copy (sığ kopy) kopyalamada sadece içerik kopyalanmasıdır yani referans kopyalanmaz.
* Deep copy kopyalamada referansın da kopyalanmasıdır.(pdf 516)

GUI PROGRAMLAMA

* setDefaultCloseOperation(EXIT\_ON\_CLOSE) pencere çarpısına basınca kodun da sonlanmasını sağlar.
* setVisitible(true) methodu en sonda olmalı. Öncesinde pencerede yazılan değişiklikleri ayarlamak için.
* setBouds() methodu JPaneli pencereye yerleştiren koordinatları ayarlamamızı sağlar.
* Sonrasında add ile eklenir.
* setLayout() butonların nasıl dizileceğini ayarlar içerisine gridLayout(2,3) verilirse 2 satır 3 sütun şeklinde dizer.
* MouseMtionListener mousun hareketlerinin vee mous konumunun takip edildiği yapıdır.

Multi-Tread Programlama

* Elimizde birden fazla görevler varsa bu görevler tanımlanmalı ve Tread sınıfından yaratılan nesnelere bağlamamız gerekiyor bu nesnelere de **runnable** nesne denir.
* UDP (packet base comminacation) de paket kaybı gerçekleşebilir
* TCP(stream base communication) de paketler gönderilene kadar gönderme işlemine devam edilir.
* Klasik socket istemciler (client) tarafında bulunur. Ve bir sunucuya bağlanmak için kullanılır.
* Server socket Sunucu tarafında bulunur. İstemcilerden gelen talepleri karşılamak için tasarlanmıştır.
* Port , her bir socketin ilişkilendirilme numarasıdır.0 ile 65536 arasındadır . 0 ve 1024 arasındaki portlar işletim sisteminin kullandığı rezerve portlardır.
* Accept() methodu o sunucuya gelen bağlantı ve izleme isteklerinin kabul eder.