Java - İş Görüşmelerinde Sık Çıkan Sorular

Facebook grubumuza katılımınızı bekliyoruz : JAVA - J2EE - ANDROID Türkiye [https://www.facebook.com/groups/480633008725200/]

1-Polimorfizm nedir ?
2-Inheritance nedir?
3-Interface ve Abstract class arasındaki farklar ?
4-Java5 ne getirmiştir ?
5-Checked ve Unchecked Exceptionlar nedir ?
6-Overriding nedir ?
7-Overloading nedir?
8-Serilization nedir ?
9-final anahtar kelimesi ne işe yarar ?
10-error ve exception farkı nedir ?
11 - Statik değerlerin özellikleri ?
12 - Encapsulation nedir ?
13 - Object oriented prensipleri nelerdir ?
14 - Coupling nedir ?
15 - Cohesion nedir ?
16 - Integer ve int arasındaki fark nedir ?
17 - Örnek bir kaç pattern anlat ? (Singleton , factory)
18 - Model-View-Controller tasarımından bahsediniz ?
19 - Waterfall proje döngüsünden bahsediniz ?
20 - Scrum nedir ? Nasıl uygulanır ?
21 - Neden java yı tercih ettiniz ?
22 - Upcasting nedir ?
23 - Downcasting nedir ?

1- Polimorfizm nedir ?

Polimorfizm tanım olarak Java nın upcasting özelliğinin kullanılarak runtime sırasında oluşturulan sınıfın metodunun çağrılmasıdır .

Şimdi adım adım örnekle açıklayalım

Yapacağımız örnekte *connectionAc* metodu tanımlayarak , runtime sırasında (çalışma anında) ilgili sınıfın metodunun çağrılmasını sağlayacağız

Adım 1 : Connection interface i tanımlayalım

```
package com.polimorfizm;

public interface Connection {

 public void connectionAc();
```

```
}
```

Adım 2 : MySqlConnection sınıfını tanımlayalım . Bu sınıf Connectiin arayüzünü implemente etsin

```
package com.polimorfizm;



public class MsSqlConnection implements Connection {



 @Override

 public void connectionAc() {

  System.out.println("MsSql connection açıldı");



 }

}
```

Adım 3 : MySqlConnection sınıfını tanımlayalım . Bu sınıf Connectiin arayüzünü implemente etsin

```
package com.polimorfizm;



public class OracleConnection implements Connection {



 @Override
```

```
 public void connectionAc() {

  System.out.println("Oracle connection açıldı");



 }

}
```

Adım 4 : Şimdi bu sınıfları kullarak connection açmamızı sağlayan util bir sınıf yazalım

Bu sınıf gelen Connection sınıfının tipine göre ilgili connection ı açar. İşte polimorfizm bu noktada başlıyor . Compile anında parametre olarak gelecek sınıfı bilmiyoruz . Ama runtime sırasında bu metoda MySqlConnection veya OracleConnection sınıflarından bir tanesi cast edilebilir(upcasting)

```
package com.polimorfizm;

public class ConnectionUtil {

 public static void connectionAc(Connection con){
  con.connectionAc();
 }


}
```

Adım 5 : Şimdi test ederek polimorfizmi görelim . Bir döngü içerisinde random sayılar üretelim . Eğer sayı 0 gelirse OracleConnection , 1 gelirse MySqlConnection sınıfı oluşturulur . conenctionAc metoduna hangi sınıfı gönderirsek o sınıfın metodu çalışır . Görüldüğü gibi compile anında ne sonuç üretileceğini bilemiyoruz , runtime sırasında karar veriliyor

```
package com.polimorfizm;

import java.util.Random;

public class Test {
```

```
 public static void main(String[] args) {

  Random rnd = new Random();

  Connection con = null;
  for (int i = 0; i < 5; i++) {
   // 0 ve 1 sayılarından bir tanesini random üret
   int randomSayi = rnd.nextInt() % 2;

   if (randomSayi == 0) {
    con = new OracleConnection();
   } else {
    con = new MsSqlConnection();
   }

   ConnectionUtil.connectionAc(con);

  }

 }

}
```

2- Inheritance(Kalıtım) nedir ?

Bir sınıfın özelliklerini onu extend eden sınıflara aktarmasıdır

Şimdi adım adım örnekle açıklayalım

Adım 1 : Bir Araba sınıfı tanımlayalım ve her arabanın ortak özelliği olan hizlan metodunu ekleyelim

```
package com.polimorfizm;



public class Araba {



 public void hizlan(){

  System.out.println("Araba hızlanıyor");
```

```
 }



}
```

Adım 2 : Araba sınıfını extend eden Mercedes sınıfı yazalım .

```
package com.polimorfizm;

public class Mercedes extends Araba {

}
```

Adım 3 : Alttaki örnekte görüldüğü üzere Mercedes sınıfı Araba sınıfının hızlan metodunu kalıtım yoluyla alarak çağırabilmiştir

```
public class Test {



 public static void main(String[] args) {



  Mercedes mercedes = new Mercedes();

  mercedes.hizlan();



 }

}
```

3- Interface ve Abstract sınıf arasındaki farklar nelerdir ?

Interface içerisinde metodların içerisi boştur fakat abstract sınıflar içerisinde implemente edilmiş(içi dolu olan) metodlar yazabiliriz

4- Java 5 ne getirmiştir ?

- Generics : Typed Collections List<String>
- Loop yapısı : for(String s : list)
- Autoboxing / unboxing : Integer int arasındaki dönüşümü jdk yapar
- Annotation : @Override
- Enum
- Static import
- Var args

5- Checked ve unchecked exception farkı nedir ?

Checked : Compile time sırasında handle etmemiz gereken exceptionlar. FileNotFoundException
Unchecked : Compiler ın handle etmeye zorlamadığı exceptionlar . NullPointerException

Örnek üzerinden gidersek bir dosyayı açmaya çalıştığımızda compiler kodlarımızı try-catch bloğuna almaya zorlar

```
public class Test {

 public static void main(String[] args) {
  try {
   File file = File.createTempFile("myfile", "txt");
  } catch (IOException e) {
   System.out.println("Dosya açma hatası");
  }
 }
}
```

Aşağıdaki kod bloğunda NullPointerException almamıza rağmen compiler kodumuzu try-catch bloğuna almaya zorlamaz

```
public class Test {

 public static void main(String[] args) {
```

```
  String s = null;
  s.charAt(1);
 }
}
```

6-Overriding nedir ?

Override : Kalıtım yoluyla miras olarak aldığımız metodu ezmektektir

Basit bir örnekle inceleyelim

```
package com.polimorfizm;

public class Araba {

 public void hizlan(){
  System.out.println("Araba hızlanıyor");
 }

}
```

Adım 2 : Araba sınıfını extend eden Mercedes sınıfı yazalım .

```
package com.polimorfizm;
```

```
public class Mercedes extends Araba {

   public void hizlan(){
      System.out.println("Mercedes hızlanıyor");
   }
}
```

Görüldüğü gibi *hizlan* metodu subclass(alt sınıf - extend eden sınıf) tarafından override edilmiştir

7-Overloading nedir ?

Overloading bir sınıf içerisinde metodu aynı isimle , aynı geri dönüş değeriyle fakat farklı parametrelerle kullanmaktır

Basit bir örnekle inceleyelim

```
package com.polimorfizm;


public class Math {


  public int toplamaYap(int i + int j){

       return i+j;

  }


  public void toplamaTap(int i + int j + int k){
     return i+j+k;


  }



}
```

8-Serilization nedir ?

Serilization ile java objeleri byte lar halinde stream lere yazılabilir . Böylece bir java objesi veritabanına kaydedilebilir

```
package com.polimorfizm;



public class Kisi implements Serializable{


   private String ad ;
   private String soyad;


}
```

9 - final anahtar kelimesi ne işe yarar ?

Final variables(değişkenler) : Değiştirilemez . Static ile beraber Constant olarak kullanılabilir

Final class :Extend edilemez

Final methods : Override edilemez

10 - error ve exception farkı nedir ?

Error :OutOfMemoryError gibi hande edilemeyen hatalardır . Fakat exception tipindeki hataları try-catch ile ele alabiliriz

11 - Static değerlerin özellikleri nelerdir ?

Static variable : Class a aittir yani her nesne için bir defa oluşturulmaz
: Memory de bir defa oluşturulur

Static method : Class a aittir yani her nesne için bir defa oluşturulmaz

Class ismiyle çağrılabilir

Static block : Class execute edildiği zaman heap de yerini alır

12 - Encapsulation nedir ?

Bir sınıf içerisindeki değişkenlerin private yapılarak bu değişkenlere diğer sınıflardan erişimin get ve set metodlarıyla yapılmasına denir

```
public class Kisi {

 private String ad;
 private String soyad;

 public String getAd() {
  return ad;
 }

 public void setAd(String ad) {
  this.ad = ad;
 }

 public String getSoyad() {
  return soyad;
 }

 public void setSoyad(String soyad) {
  this.soyad = soyad;
 }


}
```

Örnekte görüldüğü gibi ad ve soyad değişkenleri private yapılarak bu değişkenlere get... ve set... metodlarıyla erişim sağlanmıştır

13 - Object oriented prensipleri nelerdir ?

Bir sınıf içerisindeki değişkenlerin private yapılarak bu değişkenlere diğer sınıflardan erişimin get ve set metodlarıyla yapılmasına denir

- Encapsulation : Değişkenler private yapılarak get.. ve set.. metodlarıyla erişilir
- Abstraction : Gereksiz ayrıntılar kullanıcıya sunulmaz
- Inheritance : Bir sınıfın özellikleri extend anahtar kelimesiyle miras alınır.
- Polymorphism : Runtime sırasında oluşturulan sınıfın metodunun çağrılmasıdır

14 - Coupling nedir ?

İki sınıfın birbirine olan bağımlılığı diyebiliriz . Nesneye yönelik programlamada nesnelerin birbirleriyle az bağımlı olması benimsenir (low coupling) . Örnek olarak A sınıfı B sınıfına ne kadar bağımlıysa o kadar coupling(bağımlılık) artar . Bu nedenle projenin bakımı ve genişlitebilirliği azalır . Low coupling için çeşitli prensipler uygulanır : interface kullanımı, çeşitli tasarım patternleri

15 - Cohesion nedir ?

Bir sınıfın oluşturulma amacını ne kadar temsil ettiğini gösterir . Nesneye yönelik programlamada sınıfların belirli bir amaca yönelik yazılması beklenir (high cohesion). Örnek verirsek matematik işlemi yapan bir sınıfa file işlemleri yapan bir metod daha eklediğimiz zaman high cohesion prensibini kırmış oluyoruz

16 - Integer ve int arasındaki fark nedir ?

- int : primitive değişken tipidir. Sınıf olmadığı için herhangi bir metoduda yoktur
- Integer : Bir sınıftır . parseInt gibi metodları mevcuttur . Bu sınıflara wrapper sınıflarda diyebiliriz .Yani içerisinde int tipinde bir değişken ve sayısal işlemler yapmamızı sağlayan metodlar mevcuttur

16 - Örnek bir kaç yazılım patterninin anlatınız ?

- Singleton
- Factory

17 - Model-View-Controller tasarımından bahsediniz ?

18 - Waterfall proje döngüsünden bahsediniz ?

19 - Scrum nedir ? Nasıl uygulanır ?

20 - Neden java yı tercih ettiniz ?

Burada nesne yönelimli programladan ve javanın c++ ve c# dillerine göre bazı farklılıklarından bahsedilebilir.

Bir kaç tanesinden bahsedeyim :

- Java da yaptığımız geliştirmeler platform bağımsız olarak tüm işletim sistemlerinde çalıştırılabilir
- Java tasarımcıları Java yı öğrenilmesi ve kullanılması kolay bir dil olarak tasarlamışlardır
- C ve C++ programcılarının kolay bir şekilde adapte olabilir
- Pointer kullanılmaz
- Memory management işlemleriyle uğraşmayız
- Inheritance özelliği ile bir sınıf extend edildiği zaman özellikleri alınmış olur
- Encapsulation özelliği ile bir sınıf içerisinde abstraction sağlanmış olur .
- Polymorphism özelliği ile runtime sırasında ilgili metod çağrılır
- Multithreaded özelliği ile aynı anda thread ler açılarak farklı işlemler yapılabilir

Java güvenilirdir

- Compile time sırasında type kontrolü yapılır
- Error handling (try/catch/finally)
- Garbage collection
- Memory allocation

21 - Upcasting nedir ?

Bir sınıf extend ettiği sınıfa cast edilebilir (tip dönüşümü) . Şimdi konuyu daha iyi anlayabilmek için bir örnek yapalım

```
public class Kisi {


   public String ad ;
   public String soyad;


}
```

```
public class Calisan extends Kisi{

   private Double maas ;

}
```

Örnekte görüldüğü gibi Kisi sınıfı ve bu sınıfın özelliklerini kendisine kalıtım yoluyla alan Calisan sınıfı tanımlanmıştır

```
public class Test {

 public static void main(String args[]){

   Kisi kisi = new Kisi();

   Calisan calisan = new Calisan();

   kisi = calisan;     // upcasting

   // burada Calisan sınıfının referansı
                //  bir üst sınıfın(Kisi nin) referansına ataniyor

 }



}
```

22 - Downcasting nedir ?

Bir sınıf kendisini extend eden sınıfa cast edilebilir . Buna downcasting denir .

```
public class Kisi {
```

```
   public String ad ;
   public String soyad;


}
```

```
public class Calisan extends Kisi{


   private Double maas ;


}
```

Örnekte görüldüğü gibi Kisi sınıfı ve bu sınıfın özelliklerini kendisine kalıtım yoluyla alan Calisan sınıfı tanımlanmıştır

```
public class Test {

 public static void main(String args[]){

     Kisi kisi = new Calisan();

     Calisan calisan = new Calisan();

     calisan = (Calisan)kisi;     // downcasting

     // burada Kisi sınıfının referansı
     //bir alt sınıfın(Calisan) referansına ataniyor
 }


}
```

Burada nesne yönelimli programladan ve javanın c++ ve c# dillerine göre bazı farklılıklarından bahsedilebilir.

1-Polimorfizm nedir ?
2-Inheritance nedir?
3-Interface ve Abstract class arasındaki farklar ?
4-Java5 ne getirmiştir ?
5-Checked ve Unchecked Exceptionlar nedir ?
6-Overriding nedir ?
7-Overloading nedir?
8-Serilization nedir ?
9-final anahtar kelimesi ne işe yarar ?
10-error ve exception farkı nedir ?
11 - Statik değerlerin özellikleri ?
12 - Encapsulation nedir ?
13 - Object oriented prensipleri nelerdir ?
14 - Coupling nedir ?
15 - Cohesion nedir ?
16 - Integer ve int arasındaki fark nedir ?
17 - Örnek bir kaç pattern anlat ? (Singleton , factory)
18 - Model-View-Controller tasarımından bahsediniz ?
19 - Waterfall proje döngüsünden bahsediniz ?
20 - Scrum nedir ? Nasıl uygulanır ?
21 - Neden java yı tercih ettiniz ?
22 - Upcasting nedir ?
23 - Downcasting nedir ?

24 - Operatör önceliklerine göz atmak gerekir . Kağıt üzerinde bir kaç işlem sonucu çıkan çıktı sorulabilir
25 - Access type lar hakkında bilgi veriniz (public , protected , private , default)
26 - Default constructur nedir ?
27 - garbage collection ne zaman devreye girer ?

Yaptığınız bitirme projelerinden veya şirketlerde yaptığınız projelerden bahsedin ?

Facebook grubumuza katılımınızı bekliyoruz : JAVA - J2EE - ANDROID Türkiye [https://www.facebook.com/groups/480633008725200/]

Posted 9th March 2014 by Serkan Sakınmaz

Labels: Java - İş Görüşmelerinde Sık Çıkan Sorular

0 Add a comment

Enter your comment...

Comment as: Bünyamin Alaç ▼ Sign out

Publish Preview Notify me