**Suallar:**

* Design Pattern dedikdə nə başa düşürsünüz?

Design patterns təcrübəli obyekt yönümlü proqram tərtibatçıları tərəfindən istifadə edilən ən yaxşı təcrübələri təmsil edir. Design patterns proqram təminatının hazırlanması zamanı proqram tərtibatçılarının qarşılaşdıqları ümumi problemlərin həllidir.

* Singleton Design Pattern nədir?

Bu pattern yalnız bir obyektin yaradıldığına əmin olduqdan sonra bir obyekt yaratmaq üçün cavabdeh olan tək bir sinfi əhatə edir. Bu sinif, sinfin obyektini yaratmağa ehtiyac olmadan birbaşa əldə edilə bilən yeganə obyektinə daxil olmaq üçün bir yol təqdim edir.

* Proxy Pattern nədir?

Proxy modelində sinif başqa bir sinfin funksionallığını təmsil edir. Bu tip design pattern structural pattern altinda gelir. Proxy Pattern ortada durur ve bir vasiteci rolu oynayir.Meselen bizim image deye interfacemiz var, ProxyImage ve RealImage classlari bu interfaceden implement edir. ProxyImage class-i icinde RealImage classi-nin obyektini saxlayir. Siz ProxyImage-ni cagirirsiz, oda gedib RealImage-ni cagirir. Siz ele bilirsizki RealImage ile isleyirsiz ama eslinda ProxyImage ile isleyirsiz.

* Builder Pattern nədir?

bizim X adli class-imiz var ve bu class-in n qeder deyiseni var. biz isdeyirik ki, bu class immutable olsun. Bu zaman constructor-a n qeder deyiseni elave edirik ve bu class-in set metodlari olmur. Eger biz bu n qeder deyisenden yalniz 2 sini set etsek ve digerlerine mecbur null yazmaliyiq. Lakin, bu is eziyyetli oldugundan men builder patternden istifade ederim. Bu zaman builder patterinin ozu hemin yerde qalan deyisenlere null yazir ve hemin builder patterinin icinde set metodlari olur amma get metodlari olmur. Men hemin set metodlari ile deyisenleri set edirik.

* FactoryPattern nədir və abstraction və interface-in tətbiqi

Factory pattern Java-da ən çox istifadə edilən design patterns-lərindən biridir.Factory pattern-de biz object-in yaradilma mentiqini musteriye gostermeden yaradiriq ve ümumi interfeysdən istifadə edərək yeni yaradılmış obyektə istinad edirik. Meselen bizim Shape adli interfacemiz ve Rectangle, Circle, Square adli classlarimiz var ve bu classlar Shape interfacesinden implement edib ve bunlardan elave olaraq ShapeFactory adli class-imiz var. Bu ShapeFactory classimizin icinde getShape adli method var ve bu methoda String shapeType gonderirik, bu method geriye bir Shape qaytarir yeni Shape Interfacesini implement eden bir class-in obyektini. Daha sonra Main class-inda Shapefactory classimizin obyektini yaradiriq ve getShape methodunu cagiririq. bu method bize Shape obyekti qaytarir ama bilmirik hansi hansi Shape obyekti meselen bilmirik ki, geriye qayidan Circle, Square yoxsa Rectangledir. Sadece bu Shape obyektini goturub onun icindeki draw() metodunu istifade edirik.

* AbstractFactoryPattern nədir

AbstractFactoryPattern FactoryPattern-e cox oxsayir ve geriye FactoryPattern-ler return edir. Az once dedik ki, FactoryPattern-de shapeFactory geriye Shape qaytarir ve hansi class-in obyetki oldugu bilmirik. Indi tesevvur edek ki, bir class var AbstractFactoryPattern-de ve bu class geriye Factory obyekti return edir. Amma bilmirik hansi nov Factory obyektidir. Onu qaytarir ve sende hemin Factory obyektinde isliyirsen.Amma sen bilmirsen hansi Factorydi .AbstractFactoryPattern-nin disavantaji odur ki, bir Factory-ni cagirir ancaq diger Factory-e aid metodlar da onun icinde olur. Cunki AbstractFactory class-i icinde hem getShape-i hemde getColor-u saxlayir ozunde. ShapeFactory ve Color Factory, her ikisi AbstractFactory class-indan extend edir deye, her ikisi mutleq AbstractFactory-nin icindeki getColor() ve getShape() - override elemelidirler.

* SOLID principles nələrdən ibarətdir? 5 prinsipin hər birini yazın. Ay müəllim birini unutdum ee vaxtım da olmadı araşdırım demək lazım deyil. İnterview da belə cavab keçmir

S = Single Responsibility principle. Eger bir class yaradirsizsa, o class bir seye xidmet elemelidi. Bir methodumuz varsa, calismaliyiqki o method bir dene is gorsun . Meselen Person class-i duzeldirik, bu class person-un islerine baxir meslelen ad, soyad ve s. Sonra duzeldirem BankAccount class-ini, bu class bank account islerine baxir. Meselen number ve s.

O =Open Closed principle. Bir dene ele class yazmalisan ki, o class genisliye bilen olsun extends ile implements ile vs. ile. Amma o class modifikasiya olmasin. Bununda sebebi, meselen, bir class-in icine yeni method gelende, ferz edek ki, men bir class yaziram ve o class-in icinde 5 dene method var, o class-in icine 6-inci method gelende alem deyir bir-birine ve ya o class-in icindeki method-un adi,parametri deyisende alem deyir bir-birine. Ona gore de bu class-i ele yazmalisanki, sonra class-in icine gelmesede, extend ve ya implemet prinsipi ile hemin class-in imkanlarini artira bilersiz, ama direct olaraq class-in icinde deyisiklik etmek duzgun deyil . Bir class oz icindeki deyisikliklere bagli olsun, amma extentiona yeni genislenmeye aciq olsun.

L = Liskov susbtitution principle. Extend elediyin ve ya implement elediyin her hansi bir class-disa, o class-da nezer yetirin ki duzgun extend ve ya implement olsunsun, meselen rectangle teleb olunan yere square gede bilsin. eger rectangle teleb olunan yere square gede hilmiyecekse, o zaman, square-ni rectangleden extend eleme.

I = Interface Segregation principle. Musterini mecbur eleme ki, ona vacib olmuyan metodlari implemet elesin. Bu principlede Listener adli bir sey var hansi ki, FocusListener, KeyListener ve s. Meselen biz mouse ile sag click eledik ve Emeliyyat sistemi onu dinliyir ve bizi xeberdar edir, buna deyirler Listener. Inid var Listener interfacesi ve FocusListener interfacesi bu interfaceden extend edib. Bu principe-de deyir ki, arxitekturani ele qur ki, senin interfacenin butun metodlari implement etmek kimi mecburiyyeti olmasin.

D = Dependency Inversion principle. Kankretlesmeyin, Abstractlasin. Abstract classlasin demir, Abstractlasin deyir. Yeni ki, mucerret kod yazin, kimse bilmesin ki, hansi class isledir, meselen FactoryPattern kimi.Asililigi siz ozunuzden alirsiniz verirsizin sisteme. Sistem ozu, lazim olan yere lazim obyekti verir, siz bilmeden. Sizin proyyektiniz obyektlerden asilidi axi, Meselen eger shape draw edirsinizse, circle shape-sinden asilidi. Circle bizim dependency-imizdi, yeni bizim asililigimizdi. Indi b u principle deyirki asililigi ters cevirin, asililiq sizden cixsin, Sizin proyektiniz tam islesin, ama sistem bunu ozu idare elesinki hansi obyekt lazimdirsa, onu avtomatik versin size.

* Mutable class dedikdə nə başa düşürsünüz? Nümunə gətirin

Mutable classlar initialization sonra deyerleri deyisile bilir. Biz class-in metod ve deyisen kimi deyerlerini deyise bilerik. Numune olaraq Java.util.Date, StringBuilder, StringBuffer, biz ne vaxtki mutable obyektde deyisiklik etsek , yeni obyekt yaranmiyacaq, onun evezine movcud obyektin deyerlerini deyisecek.

* Immutable class dedikdə nə başa düşürsünüz? Nümunə gətirin

Eger her hansisa bir classin icindeki deryerler stabil qalirsa hec cure deyise bilmirse, bu class immutable class adlanir. Thread Safedir. Numune olaraq String class-i immutable classdir.

* Date nədir?

Qisaca zaman, sizin zamani chapa vermek isteyirsiz, Data d = new Date(); Date class-inin obyektini yaradiram ve util paketinden import edirem. Sout(d); d.getTime geriye Long qaytarir ve 1970-ci ilden bu gune kimi kecen zamanin millisaniyesidir.

* SimpleDateFormatter nədir, parse və format nədir?

Date-i cevirmek isteyirem isdediyim formata, bu zaman SimpleDateFormatter-den istifade etmeliyem ve SimpleDateFormatter-in constructoruna String parametr otururem.

SimpleDateFormatter sdf = new SimpleDateFormatter("dd-MM-yyyy HH:mm:ss.SSS");

dd = gun; MM = ay; yyyy = il; HH = 24-luk saat sistemi; hh = 12-lik saat sistemi; mm = deqiqe; ss = saniye; SSS=millisecond

String s = sdf.format(d) deyirem. sdf.formmat()- iceriye Date qebul edir ve String qaytarir.

String ss = "01-11-2022"; siz hal-hazirda bunun uzerinde emeliyyat apara bilmezsiz, Ona gore de String-i date e cebvirmek lazimdir ki, bunun uzerinde Date emeliyyatlari apara bilim. Ilk once SimpleDateFormatter-in constructoru (dd-MM-yyyy HH:mm:ss.SSS"); gerek ss("01-11-2022") e beraber olsun. Uygun olduqdan sonra parse deyirem. yeni ki, Text-i parcaliyiram, mene uygun olan formatta basa dusmeye calisiram. yeni ki, 01-gunu, 11 -ayi , 2022-ili ifade edir ve her birini yerine otuzdurmaga calisacaq ve geriye Date return edir.

Date d = sdf.parse(ss);

* Enum nədir? Bilmirsinizsə unutmusunuzsa açın oxuyun yarımçıq cavab verməyin.  
    
  <http://javacigenclik.blogspot.com/2012/03/enum-ndir-n-ucun-istifad-olunur.html>

Enumlar normal siniflərə oxşasalar da bir neçə məhdudiyyətləri vardır.Javada enum düzəltmək üçün enum keywordunden istifadə olunur.Daha sonra enum elementləri { , } arasında yazılır.Enumu classın daxilində və ya ayrıca bir class kimi düzəldib istifadə etmək olar.Əgər classın içində düzəltsəz daxili class kimi olacaq.Biz ayrı-ayrılıqda bir class kimi düzəldəcəyik. Tutaq ki, siz sadəcə aşağıdakı tərəvəzlərdən istifadə edəcəksiniz.Əgər bu tərəvəzlərdən başqa biri daxil edilərsə xəta verməyini istəyirsiz.Bu xəta ilə qarşılaşdıqda siz bunu dərhal hiss edib xətanı aradan götürəksiniz.Beləliklə programda xətalara yol verilmə hallara azalacaq. Enum Immutabledir.Type safety-dir. Enum-u extend elemek olmaz.

.public enum Terevezler{

PAMIDOR,BIBER,BADIMCAN;

}

* Enum necə elan olunur?

public enum Enums {

topla('+'), chix('-'), vur('/'), bol('\*'), UNKNOWN;

private char c;

Enums(char c) {

this.c = c;

}

* Enum classının içindəki values(), ordinal(), valueOf() methodları nə iş görür

public enum Enums {

topla('+'), chix('-'), vur('/'), bol('\*'), UNKNOWN;

private char c;

Enums(char c) {

this.c = c;

}

public class Main {

public static void main(String[] args) {

Enums top = Enums.topla;

burada top.ordinal() deyende 0 verecek top.values() deyende topla verecek. Valueof()-da adina gore size enum qaytaracaq. meselen Enums.valueOf("topla) bu geriye topla qaytaracaq.

values():Enumdakı elementləri matris şəklində toplayır.

ordinal() :Enumda mövcud olan elementin neçənci sırada yerləşdiyini müəyyən edir.

valueOf(String name) :Bu metoddan istifadə edərək həmin adlı elementin enumda olub olmadığını müəyyən edirik.

* Enum-da constructor və dəyişənlərin necə elan edə bilərik nümunə gətirin

public enum Enums {

topla('+'), chix('-'), vur('/'), bol('\*'), UNKNOWN;

private char c;

Enums(char c) {

this.c = c;

}

* Enum-da methodları necə elan edirik, nümunə gətirin

public static Enums parse(char c){

if(c == '+'){

return topla;

}

return UNKNOWN;

}

* Enum elan edəndə arxada hansı class yaranır, bu class nəyi extend edir? Bilmirsinizsə oxuyun cavab verin:  
  <http://javacigenclik.blogspot.com/2016/07/slind-enum-ndir.html>

Arxada bir class yaranir ve o class final olur. Hani ki, enum-u extend edib

* Məlumatın File-a IO ilə yazılması və oxunması necə olur? methodu yazın və methodda yazılan kodları izah edin nə nə üçün yazılıb

File-a yazma:

public static void writeIntoFile(String text, String fileName) throws Exception {

FileWriter fw = new FileWriter(fileName);//Fayla yazma isi gorur.Fayla yazma isi gorurse ve sen bunun obyektini yaratmisansa, bu mutleq harasa yazmalidi axi.

BufferedWriter bf = new BufferedWriter(fw);//BufferedWriter-in isi buffered olaraq yazmaqdi.

bf.write(text);//text-i yazir. is bitir. sonda close lemek lazimdi

* Məlumatın File-a NIO ilə yazılması və oxunması necə olur? methodu yazın və methodda yazılan kodları izah edin nə nə üçün yazılıb.

public static void writeBytes(byte[] data, String fileName) throws Exception{

Path filePath = Paths.get(fileName););//Paths class-indan isdifade ederek get metodunu cagirir ve filename-i iceri gonderir

Files.write(filePath, data);// data-ni bura yazir

}

public static byte[] readBytes(String fileName) throws Exception{

Path filePath = Paths.get(fileName);//Paths class-indan isdifade ederek get metodunu cagirir ve filename-i iceri gonderir

byte[] byteArray = Files.readAllBytes(filePath);Nio-paketinin icindeki Files class-inin readAllBytes() metodunu cagirir ve Bu geriye byte-lari return edir

return byteArray;

}

File-ler ile islemek ucun daha rahat yeni 1-2 setirlik kod ile hell elemek ucun bir imkandir

=================================================================

* FileWriter ilə BufferedWriter arasındakı fərq nədir?

Java FileWriter class-i character-oriented data-ni file-e yazmaq ucun istifade edilir. BufferedWriter ortada durur ve deyir ki, gonder mene, butun datani ozunde toplayir ve gonderir file-ye ki, yaz. Eger BufferedWriter ortada olmasa idi, butun data bir-bir file yazilacaqdi. Meselen "salam" sozu birinci yazilir file-ye. Ve bu zaman File-ni acmaq connection acilir baglanir vs .s. Kifayet qeder isler gorulur. Sonra "necesen" sozunu gonderir fayla. Hemin prosesler tekrar bas verir. Amma BufferedWriter olanda, butun data-ni ozune yigir, ve bu prosesler bir defe bas verir.

=================================================================

* Marker interface nədir?

Marker interfeysi içərisində heç bir metod və ya sabit olmayan interfeysdir. Sadece obyekti diger obyektlerden ferqlendirmek ucun istifade edilir. Meselen bir metodumuz var ve o method Obyektler qebul edir, eger biz istesek ki, yalniz A interfaceni implement eden Obyektler qebul edilsin. O zaman A interfaceni yaradiriq ve ici bos olur. Metodun parametrine de meselen public static void foo(A a){} yaziriq.

=================================================================

* Serialization interface-ini nə üçün istifadə edirik?

Java-dacSerialization bize icaze verir ki, Object-i stream-e cevirek. Hansi ki, biz network uzerinden gondere bilerik ve ya onu file kimi saxlaya bilerik ve ya onu daha sonra istifade ucun Database-de saxlaya bilerik

=================================================================

Obyekti fayla necə yazırlar və necə oxuyurlar?

public static void writeObjectToFile(Serializable object, String name) throws Exception {

try ( FileOutputStream fop = new FileOutputStream(name); //file-nin adinnan FileOutputStream duzeldir.

ObjectOutputStream oos = new ObjectOutputStream(fop)//FileOutputStream -den ObjectOutputStream duzeldir ;) {

oos.writeObject(object);//ObjectOutputStream-e deyir-ki obyekt write ele.

}

}

public static Object readFileDeserializable(String name) throws Exception {

Object obj = null;

try ( FileInputStream fis = new FileInputStream(name);

ObjectInputStream ois = new ObjectInputStream(fis);) {

obj = ois.readObject();

} finally {

return obj;

}

}

=================================================================

* Transient keyword-ü nə üçün istifadə olunur?

meselen bizim password deyisenimiz var, ve istemirik ki, onu fayla yazaq. Elan elediyin deyiseni fayla yazmamaq ucun hemin deyisenin qarsisina transient keywordu yazilir

public transient String password;

=================================================================

* serialVersionUID nə üçün istifadə olunur?

Biz User class-ini fayla yazdiq, oxuduq. indi ise User class-inin icinde neyise deyisecem. Ve yeniden bu class-i oxuyanda error verecek. cunki icinde deyisiklik olub. Bunun-da qarsisini almaq ucun, Bu class-in icine birdene deyisen yaziriq hansi ki, Useri ifade edir. Bu ifade static,long ve version ifade edir. Bunu ona gore yazirsiniz ki, eger bu class-i fayla yazdim ve sabah uzerinde deyisiklik edecemse bu class-in, ve fayl-dan yeniden elave oxumaq ucun bunu: private static final long serialVersionId = 1L; elave edirsiz.

=================================================================

* File-a yazma və ya oxuma zamanı sonda hansı obyekt close olunur və bu obyekti try-with-resources ilə automatic necə close edə bilərsiniz?

inputStream ve ya OutputStream

try ( FileInputStream fileInputStream = new FileInputStream(file);) {

}

=================================================================

* Obyekti try-with-resources ilə close edə bilmək üçün class hansı interface-i implements etməlidir

Obyekti try-with-resources ilə close edə bilmək üçün Close interfaceni implement etmelidir

=================================================================

* Bir obyekti fayla yaza bilmək üçün həmin class hansı interface-i implements etməlidir?

Serializable fayl-ini implemet etmelidi

=================================================================

* Annotation nədir, öz annotationımızı necə yarada bilərik?

Annotasiyalar proqram haqqında əlavə məlumat vermək üçün istifadə olunur.

Annotasiyalar '@' ilə başlayır.

@Target(ElementType.METHOD)

@Retention(RetentionPolicy.SOURCE)

public @interface BoshOlmasin {

}

=================================================================

* AnnotationProcessor nədir?

Annotasiyalar kompilyasiya zamanı və ya icra müddətində proqrama məlumat verir, bunun əsasında proqram əlavə tədbirlər görə bilər. Annotasiya prosessoru kodun yaradılması, xətanın yoxlanılması və s. kimi funksionallığı təmin etmək üçün bu annotasiyaları tərtib və ya iş vaxtında emal edir

=================================================================

* Reflection nədir, Reflection istifadə edərək Classın içindəki methodların adlarını necə çap edə bilərsiniz?

java.lang.reflect paketi siniflər və obyektlər haqqında əks etdirici məlumat əldə etmək üçün istifadə olunan siniflər və interfeyslər təqdim edir. Reflection API yüklənmiş siniflərin sahələri, metodları və konstruktorları haqqında məlumatlara proqramlı giriş imkanı verir. Bu, təhlükəsizlik məhdudiyyətləri daxilində əks olunan sahələrin, metodların və konstruktorların istifadəsinə imkan verir. Bu arayış sizi java.lang.reflect paketində mövcud olan sadə və praktik üsullarla aparacaq.

Method[] method = obj.getClass().getDeclaredMethods();

for (int i = 0; i < method.length; i++) {

System.out.println(method[i].getName());

}

=================================================================

* Debug rejimində işləmək dedikdə nə başa düşürsünüz və Netbeansdə bunu necə əldə edə bilərik.

bug bocek demekdir, her hansisa problem olduqda senin proqraminda bug var deyilir. Debug etab-etab applicationu yazirsan ve buglardaki temizlemeye calisirsan. Hisse-hisse baxirsanki, menim app-imde bug haradadir. Debug isaresine basirq ve stepover duymesi: hemin setri icra ele ve bu setri kec. Stepinto: Bu metodun icne gir.

=================================================================

* Ant, Maven, Gradle (Ant+Maven) - Bu gördükləriniz nədir və bunlardan hər hansısa biri olmadan biz kodumuzu compile edə bilərikmi? Əgər edə bilərik deyirsinizsə o zaman bunlar niyə lazımdır?

Pom.xml deye bir sey var, bu maven-in standartidir. Maven-in ustunluyu senin dependecy-lerin interentden yuklene biler. Biz java compileri ise saliriq, clean and build edirik, bu zaman dist ve build yaranir, buildin icinde class-lar compile olur, yeni ardicil ve s. olur. Bunu eden ya mavendi ya ant yadaki gradle. Biz compileri cagiranda eslinde Ant-a deyirik, Ant gedir compileri cagirir. Ant-in maven-e gore ustunluyu odur ki, sen ant faylinda deye bilirsen ki, evvelce bu ise dussun sonra bu. Buna deyirler compile life cyclesine mudaxile etmek. Maven-de ise deye bilmirik, Maven-da yalniz mene hansi jarlar lazimdi onu deye bilirsen. Yeni ki, Maven-in life cyclesi ozune aiddir, hec kim buna mudaxile ede bilmir. Ant-da dependency anlayisi yoxdu, ozun manul olaraq elave edirsen. Amma icinde hem dependency-si olsun hemde life cycleye mdaxile, bu gradle-dir. Bu ucune deyirler build-tools. Bunlar proqramlasdirma dili kimi bir seydi emelli-basli. Bunlardan hər hansısa biri olmadan biz kodumuzu compile edə bilərik. Sadece her birinin oz ustunluyu olduguna gore biz Ant ve ya Maven ve ya Gradle istifade etdik.

=================================================================

* Lombok nədir bizə nə kimi köməyi dəyir?

Lombok deye bir .jar file-si var. Lombok jar olaraq duzeldilib ve clean and build edilib, atilib internete. Ve men download edib, istifade edirem. Buda bir annotation processordu. Bunun isi, source kodu goturur modefikasiya edir. Biz lombok-un icinden Annotationlari goturub isdifade ede bilerik.

=================================================================

* Apache Commons nədir bizə nə kimi köməyi dəyir?

Apache Commons dependencylerden biridir. Commons yeni hamiya lazim olan gozellikleri atiblar Apache Commons Jar-inin icine. Commons-un məqsədi təkrar istifadə edilə bilən, açıq mənbəli Java proqram təminatını təmin etməkdir. Commons üç hissədən ibarətdir: proper, sandbox, and dormant. Biz apache lang3 -den istifade etdik

=================================================================

* Git nedir, subversion nedir, github, bitbucket nedir?

github, bitbucket gir protokolu ile isleyen serverdir. Subversion proyekt ile ismek ucun bir protokoldu, amma bu git deyil. subversion--da versiyalama var. Subversion sadece bir kod uzerinde islemek ucun protokoldu, sadece onun qayda-qanunlari ferqlidi.

=================================================================

* Git ilə Github arasındakı fərq nədir?

Insanlar arasinda danisigi, insan dilinden komputer diline cevrile biliner ki, insanlar bir-bir unsiyyet qurmadan, kod vasitesile ile danisa bilsinler, hemin o danisiq diline deyirler git. Git-in icnde bir nece emr var. Komputer git protokolunu tanimir, git codelerini taniya bilmesi ucun komputere git protokolu ile yazilmis server yuklenmelidi. Github ise git protokolu ile isleyen Serverdir

=================================================================

* Sıra ilə izah edin hansı nə iş görür? Init, clone, pull, push, checkout, fork

Fork=> edende kimin reposunu fork edirsense, hemin repo kopyalanir atilir senin githubina. Biz indi eyni proyekt uzerinde isleyirik, yeni kimin reposunu fork elemisense, hemin repo uzerinde bir isleyirsiz.

push => mendeki proyekti github-a gonder

pull =>Deyisiklikleri Github-dan gotur, komputere at.

clone => Git clone əmri mövcud Git repozitoriyasını kopyalayır.

checkout => Branchin yoxlanılması işçi qovluğundakı faylları həmin Brancda saxlanılan versiyaya uyğunlaşdırmaq üçün yeniləyir və o, Git-ə həmin Branchda bütün yeni commit-leri qeyd etməyi bildirir.

init =>git init əmri yeni Git repozitoriyası yaradır. Mövcud, dəyişdirilməmiş layihəni Git repozitoriyasına çevirmək və ya yeni, boş repozitoriyası işə salmaq üçün istifadə edilə bilər.

=================================================================

* TCP Protokolu dedikdə nə başa düşürsünüz?

Byte -lari sebeke uzerinde kimese ve ya harasa gonderende mueyyen protokolla gonderirisen. Hemin protokol TCCP protokolu adlanir. Bu zaman, data paketlere bolunur ve her paketde Header ve body olacaq.

=================================================================

* Java dili olmasaydı TCP anlayışı mövcud olardımı? Yəni TCP ilə işləyə bilməyimiz üçün Java dili mütləqdirmi?

TCP bir qaydadir, bunun her hansi proqramlama dili ile elaqesi yoxdur.

=================================================================

* TCP Client dedikdə nə başa düşürsünüz?

TcpClient sinfi sinxron bloklama rejimində şəbəkə üzərindən axın məlumatlarını birləşdirmək, göndərmək və qəbul etmək üçün sadə üsulları təmin edir. TcpClient-ə qoşulmaq və məlumat mübadiləsi aparmaq üçün TCP ProtocolType ilə yaradılmış TcpListener və ya Soket daxil olan əlaqə sorğularını dinləməlidir.

=================================================================

* TCP Server dedikdə nə başa düşürsünüz?

TCP serveri tanınmış portu (və ya IP ünvanı və port cütünü) dinləyir və TCP klientlərindən bağlantıları qəbul edir. TCP müştəri server ilə əlaqə qurmaq üçün TCP serverinə qoşulma sorğusu göndərir. Həqiqi TCP serveri bir rozetkada birdən çox əlaqəni qəbul edə bilər.

=================================================================

* TCP üzərində qurulmuş başqa hansı protokolları bilirsiniz?

SMPP, HTTP protokollari TCP uzerinde qurulub. SMPP => mesaj gondermek ucundu. SMPP arxada connectionu TCP ile qurur ve datani TCP ile gonderir

=================================================================

* Thread nədir?

Eyni anda bir nece is gormek ucun Thread-den istifade edeceyik. Eger bir application ise dusubse, bu o demekdir ki, bir dene thread ise dusub. Bir application en azi bir threadin icinde isliyir. Praktiki cehetden, thread demek, proses demekdi, yeni ki, is sahesi demekdir

=================================================================

* Runnable nədir?

Java-da Runnable interfeys Thread kimi işləyə bilən bir interfeysdir. Threads ilə işləyərkən, Runnable interfeys Java proqramlaşdırma dilinin əsas elementi kimi çıxış edir. Threads-i işə salmaq üçün yaradılmış Java sinifləri bu interfeysi yeni ki, Runnable interfaceni implement elemelidir.

=================================================================

* Threadləri yaradarkən extends Thread daha üstündür yoxsa implements Runnable

extends Thread eledik, biz bunnan sonra hemin class-da basqa extend elemek qabilliyetini itirdik. Cunki Her class yalniz bir defe extend ede biler. Yeni ki, biz class-imizin basqa bir class-dan extend elemek qabiliyyetini itiririk. Bu qabiliyyeti itirmemekden oteride biz implements Runnable deyirik

=================================================================

* race condition nədir və race conditionın qarşısını almaq üçün nə etməliyik?

Threadler bir yarisma icine girirler, Meselen a deyiseninin kohne deyerini goturursunuz ve onu bir vahid artirirsiz. A-nin kohne deyeri 90 di, ve men 100 dene thread buraxmisam, hamsi eyni anda a-nin deyerini goturub bir vahid artirmaq isdeyende en az 5 dene thread ust-uste dusur. Bu 5 dene thread a-ni bir vahid artirir ve a -nin deyeri olur 91. Bu adlanir race condition, yeniki eyni menbeye eyni vaxtda muraciet edirik.

Terifi: Shared resources, ne zaman ki, threadler bu shared resource-ye eyni andamuraciet edirler ve hemin menbe synchronized olunmuyub, bu zaman race condition bas verir.

=================================================================

* Synchronization dedikdə nə başa düşürsünüz?

Eyni anda buna yalniz bir nefer(thread) muraciet ede biler. Yeni ki, threadleri siralayir, bir- bir muraciet edirler threadler

=================================================================

* Public syncrhronized void foo(){} buradakı synchronized keywordü nə üçün istifadə olunur?

burada ki, foo() metoduna muraciet eden threadler, sira ile yeni ki, bir-bir muraciet edecek.

=================================================================

* monitor nədir, lock nədir, synchronized(obyekt) nə üçün istifadə olunur

public void doIt1() {

synchronized (monitor){

System.out.println("doIt1");

}

}

public void doIt1() {

synchronized (monitor){

System.out.println("doIt1");

}

}

Monitor budur. Monitor-a obyetki manage eden yeni idare eden. Eger iki dene method-un icinde synchronized(obyekt){} varsa, ve monitor eynidise, eger thread birinci methodun synchronized(obyekt){} hissesine girirse, bu zaman, ikinci metodun synchronized(obyekt){} hissesi, birinci methoddaki thread isini qurtarana kimi lock olunur

synchronized(obyekt) ona gore istifade olunur ki, methodun icinde synchronized(obyekt) yazilan yerde thredaler bir-birini gozleyir

=================================================================

* deadlock nədir və bunun qarşısını almaq üçün nə etməliyik?

deadlock iki thread-in bir-birini gozlemesidir. Thread1 deyirki Thread2 -ni gozleyirem Thread2 -de deyir ki, Thread1-i gozleyirem. Deadlock-un qarsisini almaq olmur. Deadlock-un yaranmasi ucun, serait yaratmiyasan gerek. Bu seraiti yaratmamaq ucun gerek monitor-un senin threadinin icinde olsun ve private olsun, yalniz senin thread-in ucun istifade edilsin. Local lockdan isdifade edin global lockdan isdifade elemeyin.

=================================================================

* volatile nə üçün istifadə olunur?

Threadler yarananda proyektin cari hali kopyalanir, yeni ki, proyektin yaddasi kopyalanir. Bu o demekdir ki, het threadin oz yaddasi olur. Bezen synchronized yazsan bele, a deyiseninin deyerini artirib elemisiz 100, threadlerden biri bunu 100 kimi digeri 0 kimi gorur, yeni ki, bir thread diger yaddas sahesinden xeberi yoxdu. Volatile yaziriq ki, bu yaddas sahesi diger threadlere melum olsun, a-nin deyeri deyisse haminin xeberi olsun.

=================================================================

* Threadlərdə join nə üçün istifadə olunur

Threadleri bir-birine joine edirsen ve bir thread diger threadi gozdeyir. Start()- cagiranda o gozlemir ki, isini qurtarsin kecsin o biri start()-a. Join yazdiqda ise gozleyir ki, birinic thread isini gorsun, sonra ikinic senra ucuncu ve s.

Thread t = new Thread(new MyRunnable1());

Thread t1 = new Thread(new MyRunnable1());

Thread t2 = new Thread(new MyRunnable1());

t.start();

t.join();

t1.start();

t1.join();

t2.start();

=================================================================

* ExecutorService nədir? Hansı növ executor service-ləri tanıyırsınız?

Threadlere nezaret etmek ucn API-dir. Executors deye bir class var ve hansi ki, geriye ExecutorService qaytaran methodlar saxlayir .

ExecutorService service = Executors.newSingleThreadExecutor();// her defe bir dene thread ise sal

service.submit(new MyRunnable1());//submit ozu icerde thread yaradib icine runnable gonderecek

Executors.newFixedThreadPool(5);bu sene demirki eyni anda bir dene thread ise sal. Sadece deyir ki, sen mene 10 denede request gondersen, men onun 5 denesini run ede bilecem.

=================================================================

* invokeAll, invokeAny, shutdown, awaitForTermination hər birini ayrı-ayrılıqda izah edin. Ay müəllim bir dənəsini unutdum eee deməyin araşdırın yazın.Dərsdə izah olunub. Dərsə baxdığınız zaman qeyd etdiyiniz notelarınıza baxın orada tapacaqsınız

shutdown => sizin service yaradan threadleriniz sondurur ki, menim sennen(thread) ile isim bitdi

ex.awaitTermination(5, TimeUnit.SECONDS);=> awaitTermination(5, TimeUnit.SECONDS)-dan sonra olan kodlar 5 saniye sonra icra olunacaq

invokeAll, invokeAny => bunlarin her ikisi sizden collection isdiyir. Ama hemin collection runnableden yox callasble isdiyir ve callable olan butun elementleri run edecek. invokeAll=> step by step run edir. invokeAny=> ordan burdan run edir

=================================================================

* Callable interface-i nə üçün istifadə olunur və Future nədir?

Tapşırığı bağlamaq və onu asinxron icra üçün Thread və ya Thread pool-a ötürmək üçün istifadə edilən mövcud Runnable interfeysini tamamlamaq üçün Java 5-də çağırıla bilən interfeys əlavə edilmişdir. Zəng edilə bilən əslində dəyəri Gələcək obyekti vasitəsilə mövcud olan asinxron hesablamanı təmsil edir.

Future is used for storing a result received from a different thread, whereas Callable is the same as Runnable in that it encapsulates a task that is meant to be run on another thread.