İlk sorunuz ---> Soru1: public static void main (String [] args) Java’da yer alan bu ***YAPI***yı açıklayınız. (edited)

Cevap1--> Bu, herhangi bir Java programı için giriş noktasıdır.  
**public**: bu yönteme kimin erişebileceğini belirtmek için kullanılan bir erişim değiştiricidir ve bu yöntemin herhangi bir sınıf tarafından erişilebilir olacağı anlamına gelir.  
**static**: Java'da sınıf tabanlı olduğunu tanımlayan bir anahtar kelimedir main (), bir sınıfın oluşumunu oluşturmadan erişilebilmesi için Java’da static yapılır.  
**void**: methodun return type dır.  
**main()**: JVM tarafından aranan methodun adıdır.  
**JVM**: Java sanal makinesi String args ise main methoda iletilen parametredir.  
:sparkle: Aciklama 1: Main metodu programin ana metodu olarak gecer. Cogu dilde bu boyledir. Main metodlarinin farkli sekillerde yazildigi farkli diller mevcuttur. Ana metodlar olmadan programin giris bolgeleri belirlenemez.Arabanın motoru, insanın kalbi, kodları calıstıracak anahtar gibi düşünebilirsiniz..

Soru2-->**AGİLE**VE**WATERFALL**METHODOLOJİ ARASINDAKİ FARKLAR NELERDİR?

Cevap2-->**Agile** methodoloji 2-4 haftalık sprintler halinde bütün aşamaların birbiri ve stockholder ile temas halinde olduğu yazılım yaşam döngüsü sürecidir. **Waterfall**metodolojide bir aşama bitmeden diğer aşamaya geçilmez. Bu konuda oldukça katı olan bu metdolojide hataların teşhisi oldukça geç gerçekleşebilir. Ayrıca stockholderın olmasını beklediği sonuç, projenin yapım aşamasındaki iletişim eksikliğinden kaynaklı aksi netice ortaya çıkarabilir. Oysa agile metodolojide sürekli takip ve projeyi sprintlere bölerek yürütme gerçekleştiğinden olası sorunlara çok hızlı müdahale edilir ve projenin daha sağlıklı bir şekilde ilerlemesi sağlanır.  
**Waterfall**modeli baştan sona her ayrıntıyı adım adım ve uzun vadeli planlarken; **Agile** modelinde kısa vadeli planlar söz konusu olur. Çünkü **Agile** yaklaşıma göre proje süreci devam ederken değişiklik yapılabilir. Her aşama **Waterfall** modelinde olduğu kadar net değildir. **Waterfall** incelemelerinde dokümanlar ön plandadır. Her aşamanın tamamlanmasından sonra incelemeler yapılır.**Agile** modelinde analizler, müşterilerle yapılan iletişim neticesinde sık sık gerçekleşmektedir. **Waterfall** modelinin uzun vadede maliyeti artırma riski vardır. **Agile** ise ürüne dair en iyi versiyonu bekler. Bu nedenle de ürünle ilgili potansiyeli zorlar ve görece daha düşük riskler taşır.**Agile** yaratıcılığa imkân tanıyan esnekliğe sahipken **Waterfall**modelinde bundan söz etmek pek mümkün olmaz. Bu noktada en başta alınan kararlara bağlılık söz konusudur.

Soru3-->JAVA’DA**INHERITENCE**’IN FAYDALARI NELEDİR?

cevap3-->OPP konseptin temel taşlarından biri olan inheritence, parent child ilişkisi çerçevesinde geliştiriciler için;

* Daha az kod yazma (less code)
* Bir class’da bir kez yazarak birçok yerde kullanma (reusability)
* Update edebilme kolaylığı (maintenance)
* Kod sadeliği ile classlar arası organizasyon sağlama (well organization)

Gibi birçok açıdan kolaylık sağlar.  
**NOT:** *İnheritence’da classlar arası ilişki aşağıdan yukarıya doğrudur. Yani child class verileri Parent’tan alır. Parent asla child class’dan veri almaz.*

Soru4--> Java'daki exception hakkında bilgi verir misiniz.(Ne tür exception lar vardır nasıl handle edilir vs..) Bildiginiz birkacını acıklar mısınız?

Exception, kod blokları arasında oluşan anormal duruma denir. Örnek vermek gerekirse bir sayının sıfıra bölünmesi, kod bloğunun noktalı virgül ( ; ) ile kapatılmaması, String bir değer döndüren data type'ına int bir data type girilmesi gibi nedeler exception'a sebep olacaktır. Basit hatalar eksik bilgilerin girilmesi, ya da yanlış girilen bilgilerin düzeltilmesi ile handle edilebilecekken, sıfıra bölünme "arithmeticException" gibi exception sorunlarında try catch bloğu kullanmak gerekir.

Aşağıdaki tabloda görüleceği üzere Java’da iki tür exception mevcuttur. Kontrolsüz olanlar (UncheckedExceptions) ve Kontrollü olanlar (CheckedExceptions). Kontrolsüz exceptionlar aynı zamanda run time exception olarak da bilinirken kontrollü exceptionlara compile time exception da denir. Aşağıdaki tabloda görüldüğü gibi “Errors” ise exception’ların ata classı olan Throwable’dan gelmektedir.

NOT: Java'da bütün hata türlerinin ata class'ı "Throwable" sınıfıdır.

**Hata (Error) :** Ölümcül bir hatayı işaret eder ve telafisi çok zordur. Örneğin OutOfMemoryError (Yetersiz Bellek) hatası oluşmuş ise uygulamanın buna müdahele edip düzeltmesi olanaksınızdır.  
**Kontrolsüz İstisnalar (UncheckedExceptions) :** Bu istisna tiplerine Çalışma Anı İstisnaları da (Run-Time Exceptions) denilir. Çünkü çalışma anında meydana gelen istisnalardır. Eğer uygulama normal seyrinde giderse ortaya çıkmaması gerekir. Örneğin, **ArrayIndexOutOfBoundsException** istisna tipi, bir dizinin olmayan elemanına eriştiğimiz zaman ortaya çıkar. Yani kontrolsüz kodlamadan dolayı meydana gelen istisna tipleridir. Java bu tür istisnalar için önceden bir önlem alınmasını şart koşmaz; yine de önlem almakta özgürsünüzdür.  
**Kontrollü İstisnalar (CheckedExceptions) :** Bu istisna tiplerine Derleme Anı İstisnaları da (Compile-Time Exceptions) denilir. Çünkü derleme anında ide’ler tarafından uyarılırız. Eğer derleyici derleme zamanında exceptionlar için try-catch bloğu göremezse hata verecektir ve kodumuz biz handle edene kadar derlenmeyecektir. Bu istisnalar çevresel koşullardan dolayı oluşabilirler. Örneğin erişilmek istenilen dosyanın yerinde olmaması (FileNotFoundException) veya ağ (Network) bağlantısının kopması sonucu ortaya çıkabilecek olan istisnalardır. Bu istisnalar için önceden önlem alınması gereklidir.

1. NumberFormatException: Format dışı kullanımında gerçekleşen durumdur. Örneğin sayı girilmesi gereken yere karakter girilmesi gibi. = compile time

2. ArrayIndexOutOfBoundsException: Array'da aranan index dışında bir değer girilmişse bu exception ortaya çıkar. = run time

3. FileNotFoundException: Oluşturulmamış bir dosyaya erişim durumunda meydana gelen hatadır. = compile time

NOT: Java böyle bir durumda bize altı kırmızı çizili bir şekilde uyarı verir ve bizden method signature'ına throws keyword'lü bir exception yazmamızı ister. Bu şekilde ilgili exception handle edilmiş olur.

4. VirtualMachineError: JVM'nin çalışmasını etkileyen durumları inceler. = Error

5. AWTError: Grafik arayüze ait hataları inceler. = Error

6. ArrayStoreException: Array'a kendi türü dışında bir veri girilmesi durumunda gerçekleşir. = run time

7. OutOfMemoryError: Bellek yetersizliği durumlarını inceler. = Error

8. ClassNotFoundException: Olmayan bir class'a erişme istediği durumlarını inceler.

= run time

9. IOException: Giriş çıkış işlemlerindeki istenmeyen durumları inceler. = compile time

10. ClassCastException: Herhangi bir nesne değişkenine farklı bir tip değer girilmesi = run time

Örnek

Object i = Integer.valueOf (42);

String s = (String)i; // ClassCastException thrown here.

11. StringIndexOutOfBoundsException: String'de var olmayan bir indekse erişilmeye çalışıldığında gerçekleşen hata. = run time

12. AritmeticException: Sıfıra bölme gibi mantıksal matematik hatalarında oluşur. Örneğin = run time

( İnt x = 10;

İnt y = 0;

System.out.println (x/y) ;

Komutu exception fırlatır. )

13. NullPointerException: Herhangi bir nesne değişkenine ilk değer atanmadan kullanılmaya çalışıldığında ortaya çıkar. = run time

14. IllegalArgumentException: Metotlara geçersiz argüman atamalarında fırlatılır.= run time

NOT: Compile Time Exceptionları kontol altına almak için try catch bloğu kullanırız.

Örnek syntax:

try {

int a= 42 / 0;

} catch (ArithmeticException Hata) {

System.out.print("Hata oluştu :" + Hata.getMessage());

Soru5-->**JAVA’DA DATA TYPE’LAR HAKKINDA BİLGİ VER**

Data type’lar primitive ve Non-primitive olmak üzere temelde ikiye ayrılır. Primitive data type’lar char, boolean, byte, short, int, long, double ve float olmak üzere sekiz tanedir.  
**Primtive Data Type’lar;**  
**boolean;** true ya da false döndürür.  
**char;** tek tırnak ‘ ’ içinde özel karakter, sayısal değer ya da bir harf alabililir. Ancak sadece bir karakter alma özelliğine sahiptir.  
**byte;** Sayısal değerler içinde en dar çerçeveye sahiptir. 8 bit hafızası vardır.  
**short;** Sayısal değer alan primitive data type’lardandır. 16 bit hafızası vardır  
**int;** En yaygın kullanılan ve tam sayı değeri alan data type’lardandır. 32 bit hafızaya sahiptir.  
**long;** tam sayı değeri alan bir diğer sayısal data type’lardandır ve 64 bit hafızaya sahiptir.  
**double;** Ondalıklı sayılar için kullanılır. 64 bit hafızaya sahiptir. Kullanımı float’a göre daha yaygındır  
**float;** 64 bit hafızaya sahiptir. Ondalık sayılar için kullanılır. Bu data type’ı kullanırken sayısal verinin sonuna f harfi konulur.  
**Non-Primitive Data Type’lar;**  
**String;**double quat içinde her türlü metni yazabildiğimiz bir data türüdür. İmmutable’dır. Yani değiştirilemez bir yapıya sahiptir. Datası belleğin Heap memory kısmında tutulur. Referansı da stack memory de bulunur. Object sınıfına bağlıdır ve bir classı vardır.  
Non-Primitive Data Type’lar oluşturulan her obje için geçerlidir ve primitive data type’lar gibi sınırlı değildir. İstediğimiz kadar Non-Primitive data type üretebiliriz. Object sınıfına bağlı olan tüm data type’lar birer nonprimitive data türüdür.

Soru6-->:point_down:  
**ELİNİZDE BİR LİSTE VAR VE BU LİSTEDEKİ TEKRARLI ELEMANLARI SİLMEK İSTİYORSUNUZ NE YAPARSINIZ?**

Cevap6-->:point_down:  
Listenin elemanlarını Collection’lardan olan Set’e atarım. Ancak List’deki elemanlar henüz tekrarlı bir liste halinde olduğu için bu defa Set içinde yer alan tekrarsız elemanların olduğu data’yı yeni bir List oluşturarak bunun içine atar ve kalıcı bir şekilde tekrarsız bir List elde etmiş olurum.

**FİNAL KEYWORD’Ü HANGİ İŞLEMLERİ YAPMAK İÇİN KULLANILIR?**

Cevap7-->:point_down:  
Değeri sabit olan ve değiştirilmesine gerek duyulmayan dataların hataen değiştirilmesinin önüne geçmek için final keyword’ünü kullanabiliriz. Örneğin pi sayısının değeri (3,14) sabittir ve değişmez. Bunun gibi datalar için final kullanılabilir.  
Bir class final olarak tanımlanırsa bu classtan inherit yapılmaz.

Method final yapılmışsa override edilemez.

Parametre final olarak tanımlanmışsa bu parametrenin değeri değiştirilemez.

Aynı şekilde yukarıda ifade edildiği gibi bir variable final olarak tanımlanmışsa daha sonra değeri değiştirilemez

**NOT:** Normalde bir variable’ı tanımlayıp daha sonra değer ataması yapabilirken. Final olarak belirlenmiş bir variable’da bunu yapamayız. Değer ataması tanımlama ile birlikte yapılmalıdır.

**Soru8--> SIZCE  BİR TESTER’IN SORUMLULUKLARI NELERDİR?**

Cevap8-->:point_down:

* Test planın takip etmek
* Hataları tarafsız ve gerçekçi bir şekilde raporlamak
* Yazımlımcıyı değil yazılımı test ettiğinin bilincinde olmak
* Riskleri tarafsız bir şekilde değerlendirmek
* Hataları önceliklendirmek
* Gerçekleri paylaşmak

**@channel Soru9-->  JAVA’DA GARBAGE COLLECTOR’UN ÇALIŞMA PRENSİBİ HAKKINDA BİLGİ VEREBİLİR MİSİNİZ?**

Cevap9-->:point_down:  
Garbage collector gereksiz nesnelerin ve bilhassa null olarak tanımlanmış olanların toplanarak hafızada yer açılması işlemidir. Bu sayede gereksiz bilgiler temizlenerek çalışmanın hızlanması sağlanır. Garbage collector herhangi bir komut verilmeden de otomatik olarak çalışabilir. Ancak biz de kod yazarak **Runtime.getRuntime.gc();** garbage collector’un devreye girmesini sağlayabiliriz. Bir şekilde biriken nesneler OutOfMemoryException hatası verebilir. Ayrıca şunu ifade etmek gerekirse garbage collector heap memory’de temizlik yapar ve JVM (Java Virtual Machine)’nin kontrolünde çalışır.

**Soru10--> SQL’İN TARİFİNİ YAPAR MISINIZ?** (edited)

Cevap10-->:point_down:  
Açılımı Yapılandırılmış Sorgu Dili (**Structured Query Language** ) olan SQL, bir veri tabanı uygulamasıdır. SQL ile veri tabanındaki verileri getirebilir, update edebilir, değiştirebilir ya da silebiliriz. Böylece SQL, database’deki bilgilerin sağlıklı bir şekilde yönetilmesini sağlar. Veri tabanına ihtiyaç duyan her web uygulaması SQL’e de ihtiyaç duyacaktır. SQL’den başka sorgulama dilleri olmasına rağmen SQL’in ücretsiz olması, açık kaynaklı bir sorgulama dili olması ve kolay anlaşılır bir syntax’e sahip olması gibi avantajları onun muadillerine göre (PL/SQL, TCL, T-SQL) daha fazla tercih edilmesine neden olmaktadır.

**Soru11-->  SQL KOMUTLARINI KULLANABİLECEĞİMİZ İDE VEYA EDİTÖRLER HANGİLERİDİR?**

PostGre SQL, MySQL, Oracle SQL, Microsoft SQL server