===SCOPE 1===İNSTANCE VARİABLE===

public class Scope1 {  
 int sayi; *// class'in icinde fakat main method'un disinda olusturulan  
 // ve static olmayan variable'lara INSTANCE (Object) variable denir  
 // class level'da olusturuldugu icin class'in her yerinden belli sartlarla kullanilabilir  
 // Instance variable'lar object'e bagli olarak calisir (ogretmen adi veya, ogrenci notu gibi)* String isim="Mehmet";  
 String soyisim; *// default deger null* boolean izindeMi; *// deafult deger false* char ilkHarf; *// default deger bosluk* public static void main(String[] args) {  
 *// System.out.println(sayi); // static olmayan bir variable'a static method icinden ulasamayiz  
 // main method'umuz static oldugu icin main method icerisinden  
 //method(); // static olmayan variable veya method'lara direk ulasamayiz  
 // instance bir variable'a main method icerisinden ulasmak istedigimizde  
 // OBJECT olusturmaliyiz* Scanner scan = new Scanner(System.*in*); *// bu javadaki standart obje   
 //class ismi obje ismi Obje olusturmada keyword class ismi () // oluşturma formudur* Scope1 obj1 = new Scope1();  
 System.*out*.println(obj1.sayi); *// buna deger atamadik java default deger verdi 0* System.*out*.println(obj1.isim);  
 System.*out*.println(obj1.soyisim); *// buna deger atamadik java default deger verdi null  
 // instance variable'lar olusturuldugunda biz istersek deger atayabiliriz  
 // Instance variable'lara eger biz deger atamazsak Java Default deger atar* Scope1 obj2 = new Scope1();  
 obj2.isim="Muslum";  
 obj2.soyisim="Baba";  
  
 System.*out*.println(obj2.isim + " " + obj2.soyisim); *// Muslum Baba* System.*out*.println(obj1.isim + " " + obj1.soyisim); *// Mehmet null* System.*out*.println(obj2.izindeMi); *// false* System.*out*.println(obj2.ilkHarf); *// space* Scope1 obj3 = new Scope1();  
 obj3.isim="Ferdi";  
 obj3.soyisim="Tayfur";  
 obj3.izindeMi=true;  
 System.*out*.println(obj3.isim +" " + obj3.soyisim+ " " + obj3.izindeMi ); *// Ferdi Tayfur true*

}  
  
 public static void staticMethod() {  
 }  
 public void method() {  
 System.*out*.println(sayi); *// bu method static olmadigi icin instance variable'lara direk erisebilir* sayi=sayi+20; *// ve bu method iceriisinde gecerli olmak uzere degerini degistirebilir*

***===SCOPE2===* public class Scope2 {  
 public static void main(String[] args) {** *// farkli class'lardan da Scope1 class'ina object olusturarak ulasabiliriz***Scope1 obj4 = new Scope1();  
 System.*out*.println(obj4.isim); //Mehmet  
 System.*out*.println(obj4.soyisim); //null  
  
 obj4.soyisim = "Can";  
 obj4.isim = "Ahmet";**

**System.*out*.println(obj4.isim + " " + obj4.soyisim); // Ahmet Can**

**-------------------------------------------------------------------------------------------------------------------------------------------------------------**

Bu sayfadaki 2 class birbiriyle bağlantılı, 2. Class a 1. Den veri ve metod çağırılıyor

**===STATİC VARİABLE===**

1.class--------

public class Scope1 {  
 static int *okulId* = 1201;  
 static String *okulAdi* = "Yildiz Koleji";  
 static boolean *acikMi*;  
  
 *// static variable'lar class level'da olusturuldugu icin class'in her yerinden kullanilabilirler  
 // instance variable'lar static olmayan method'larda direk kullanilabiliyorken, static method'larda  
 // object olusturularak kullanilabiliyordu  
 // static variable'lar ise ister static isterse static olamayan tum method'lardan direk kullanilabilir* public static void main(String[] args) {  
  
 System.*out*.println(*okulId* + " " + *okulAdi* + " " + *acikMi*); *// 1201 Yildiz Koleji false  
 okulId* = 1202;  
 *acikMi* = true;  
  
 *staticMethod*();  
  
 System.*out*.println(*okulId* + " " + *okulAdi* + " " + *acikMi*); *//1203 yıldız koleji true* }  
  
  
 public static void staticMethod() {  
 System.*out*.println(*okulId* + " " + *okulAdi* + " " + *acikMi*); *// 1202 Yildiz Koleji true  
 okulId* = 1203;  
 }  
  
  
 public void method() {  
 *okulId* = 1205;  
--0—0—0—0—0--  
**2.class-------**

public class Scope2 {  
 public static void main(String[] args) {  
  
 *// Static variable'lar icin object olusturma ihtiyaci yoktur  
 // Baska bir Class'dan static variable'lara ulasmak istedigimizde  
 // Ulasmak istedigimiz Class'in adi . static variable adi yazmamiz yeterlidir.* System.*out*.println(Scope1.*okulAdi*); // Yıldız Koleji…(Scope 1 main deki ilk bilgileri verir)  
 System.*out*.println(Scope1.*okulId*); *// 1201……(Java Run Time program'dir)  
 // Scope2 Class'i calistirildiginda Scope1 calismazzzzzz  
 // dolayisiyla en basta olusturulan veya atanan degerler gecerli olur* Scope1.*okulAdi*="Mehmet Koleji";  
 System.*out*.println(Scope1.*okulAdi*); *// Mehmet Koleji* Scope1.*staticMethod*(); //diğer clastaki staticMethod un tamamını çağırmış oldu.  
 (//işte method bu🡪 System.*out*.println(*okulId* + " " + *okulAdi* + " " + *acikMi*); *) okulId* = 1203;

*// burada cevap diğer classtaki ilk girilen cevaplar olur, sadece bu classta okul adı Mehmet Koleji diye değiştirilmişti, onu buradaki gibi atar, yani çıktı🡪 (1201, Mehmet Koleji null)*

System.*out*.println(Scope1.*okulId*); *//1203 (bir üstte 1. Classtan çağırılan staticMetodun içinden okulId değişikliği geldi o yüzden çıktı 1203)* System.*out*.println(Scope1.*okulAdi*); //Mehmet Koleji

===LOCAL VARİABLE===

public class Scope4 {  
 public static void main(String[] args) {

int sayi=10;*// 1- bir method icinde olusturulan variable, sadece o method icinde kullanilabilir  
 // 1- System.out.println(isim);* int sayi2; *// 2- bir local variable deger atanmadan da olusturulabilir  
 // 2- System.out.println(sayi2); //2- ancak ilk deger atamasi yapilmayan variable'lar kullanilamaz  
 // 2- sayi2++; // 2- ilk deger atanmadigi icin artirma yada azaltma da yapamayiz  
 // 2- Java deger atamasi olmadan local variable oluturulmasina izin verir, ilerde deger atanacak diye bekler* sayi2=15; *// 2- olusturma ayri satirda(11.satir) , deger atamasi ayri satirda (16 satir) yapilabilir  
  
 // 3- birden fazla method'un oldugu class'larda her method'da kullanmamiz gereken  
 // 3- ortak variable'lar varsa class level'da variable olusturmaliyim  
 // 3- ortak variable class'in yapisina bagli olarak instance veya static olabilir* }  
 public static void staticMethod() {  
  
 String isim="Hasan";  
 *//1- System.out.println(sayi); bu kurala main method icerisinde olusturulan variable'lar da dahildir* }  
 public void method() {  
  
 boolean isTrue=true;  
 *//1- System.out.println(sayi); Bu kural static olan veya olmayan tum methodlar icin gecerlidir*

**===LOOP SCOPE VARİABLE ===**

public class Scope5 {  
 public static void main(String[] args) {  
  
 for (int i = 0; i < 5; i++) {  
  
 String isim= "Ayse";  
 System.*out*.println(i + " " + isim);  
 }*// System.out.println(isim);// Loop icinde olusturulan variable'lar loop'a ozeldir  
 // ve loop disinda kullanilamaz* for (int i = 0; i < 5; i++) {  
 System.*out*.print( i + " ");  
 } *// System.out.println(i);* int count=0; *// method'a ait local variable'dir  
 // deger atamadan kullanmaya calisirsaniz CTE alirsiniz* while(count <5) {  
 System.*out*.println(count);  
 count++;

-----------------------------------------------------------------------------------------------------  
 public class Scope6 {  
 int num1;  
 String name = "Ali";  
  
 public static void main(String args) {  
 *add*(); *// product (5); static olmayan bir method static olan main method icerisinden cagrilamaz* }

public static void add() { *// num1++; // static olmayan (instance) num1, static bir method icinden kullanilamaz* int num2 = 6;  
 char letter;  
 System.*out*.println("Do addition ");  
 }

public void product(int num3) {  
 name = "Veli";  
 *// num2++; // num2 ustteki methodda olusturulmus Local bir variable'dir.*

===CONSTRUCTOR===

public class Car {  
 String marka;  
 String model;  
 int yil;  
 boolean kazasiVarMi;  
  
 public static void main(String[] args) {  
  
 Car car1= new Car();  
 *// Bir object olusturuldugunda esitligin saginda mutlaka Constructor kullanilir  
 // Constructor ismi clas ismi ile ayni olmalidir  
 // Constructor'da mutlaka () olmalidir  
 // Constructor'da parametre olmasi istege baglidir* System.*out*.println(car1.marka + " " + car1.model + " " + car1.yil + " " + car1.kazasiVarMi);  
 *// null null 0 false* car1.marka="Toyota";  
 car1.model="Corolla";  
 car1.yil=2010;  
 car1.kazasiVarMi=false;  
 System.*out*.println(car1.marka + " " + car1.model + " " + car1.yil + " " + car1.kazasiVarMi);  
 *// Toyota Corolla 2010 false* Car car2= new Car();  
 car2.marka="Opel";  
 car2.model="Astra";  
 car2.yil=2015;  
 car2.kazasiVarMi=true;  
 System.*out*.println(car2.marka + " " + car2.model + " " + car2.yil + " " + car2.kazasiVarMi);  
 *// Opel Astra 2015 true* System.*out*.println(car1.marka + " " + car1.model + " " + car1.yil + " " + car1.kazasiVarMi);  
 *// Toyota Corolla 2010 false* car1.kazasiVarMi=true; *// car1'e ait her hangi bir ozelligi degistirebiliriz* System.*out*.println(car1.marka + " " + car1.model + " " + car1.yil + " " + car1.kazasiVarMi);  
 *// Toyota Corolla 2010 true*

===KALIPHANE VE RUNNER===

public class CarUret { ( KALIPHANE)  
 *// SORU ...... CONSTRUCTOR BU CLASS'DA NEREDE ? su anda bu class'da gorunur bir constructor yok.  
 // bir class olusturuldugunda Java bu class'dan obje utretilecegini bilir ve gorunmeyen*

*//DEFAULT CONSTRUCTOR'i class'a yerlestirir  
 // default constructor parametresizdir dolayisiyla sadece hic bir ozelligi tanimlanamayan objeler uretir (tisort uret demek gibi)  
 // default constructor ile uretilen bir objenin tum ozellikleri sonradan tanimlanmalidir*

*// eger biz sonradan bir constructor yazarsak, Java default constructor'i iptal eder*

*// Bir constructor olusturalim* public CarUret(){  
 }  
 *// kurallar 1- ismi class ile ayni olmalidir (dolayisiyla buyuk harfle baslar)  
 //2- Constructor return type'a sahip degildir  
 // 3- constructor isminden sonra mutlaka parantez olur () ama parametre olmasi opsiyoneldir  
 // 4- bir constructor olusturuldugunda kimlerin kullanacagini belirlemek icin access modifier yazilir* String marka;  
 String model;  
 int yil;  
 boolean kazasiVarMi;  
 *// bir programda cok fazla uretecegimiz objeler icin bir tane class oluştururuz bu class'da olusturulacak objelere ait tum ozellikler olur, bu class direk calistirilmayacagi icin main method olmasa da olur (Kaliphane gibidir)* public void yakit(String yakit) { *// method* System.*out*.println("Araba yakit olarak " + yakit + " kullanir");  
 }  
 public void vites(String vites) { *// method* System.*out*.println("Araba " + vites + " viteslidir");  
 }

-----------------------------------------------

(RUNNER)----

public class CarUretRunner {  
 public static void main(String[] args) {  
 CarUret car1 =new CarUret();  
 car1.model="Corolla";  
 car1.marka="Toyota";  
 car1.yil=2010;  
 car1.kazasiVarMi=true;  
 System.*out*.println(car1.marka + " " + car1.model + " " + car1.yil + " " + car1.kazasiVarMi);  
 car1.yakit("Benzin"); //sorular CarUret class ında üretildi, cevabı burda belirtip run edince cevaplar, önceki classtaki sorularıyla burada çıktı  
 car1.vites("duz");

//Toyota Corolla 2010 true

//Araba yakit olarak Benzin kullanir

//Araba duz viteslidir

public class Otomobil { ===KALIPHANE===  
 public Otomobil(String renk){ System.*out*.println("Araba " +renk+ " renklidir");  
 *// istersek constructor'i parametreli olarak olusturabiliriz* }  
 *// biz bir constructor olusturdugumuzda java default olani yok eder  
 // dolayisiyla biz bir constructor olusturdugumuzda mutlaka  
 // default constructor'in yerine de bir constructor yazmaliyiz* public Otomobil() { *// default constructor'in yerine bunu yazmaliyiz* }  
String marka;  
 String model;  
 int yil;  
 boolean kazasiVarMi;  
  
 public void yakit(String yakit) { *// method* System.*out*.println("Araba yakit olarak " + yakit + " kullanir");  
 }  
  
 public void vites(String vites) {

System.*out*.println("Araba " + vites + " viteslidir"); }

*-----------------------------------------*   
===RUNNER===  
 public class OtomobilRunner {  
 public static void main(String[] args) {  
  
  
 Otomobil oto1 = new Otomobil(); *// default constructor kullandik* oto1.vites("otomatik");  
  
 Otomobil oto2 = new Otomobil ("Kirmizi");

//sorular kalıphanede (soutv) oluşturuldu, cevaplar burada yazıldı, run edince cevap burada çıktı, önce vites sorusu için sonra //renk sorusu için cevap yazıldığı için aynı sırayla çıktı, yakıt için cevap girilmediğinden onun cevabı çıkmadı

//Araba otomatik viteslidir

//Araba kırmızı renklidir

===KALIPHANE===

public class ParametreliConstructor {  
 String marka;  
 String model;  
 int yil;  
 boolean kazasiVarMi;  
  
 public ParametreliConstructor(int yil) { *//1. Soru* this.yil=yil;  
 }  
 public ParametreliConstructor() { *//2. soru* }  
 public ParametreliConstructor(String Marka,String Model) { *//3. soru* this.marka=marka;  
 this.model=model;  
 }  
 public ParametreliConstructor(String marka, String model,int yil, boolean kazasiVarMi) { *//4. soru* this.marka=marka;  
 this.model=model;  
 this.yil=yil;  
 this.kazasiVarMi=kazasiVarMi;  
 }  
  
 public void yakit(String yakit) { *// method* System.*out*.println("Araba yakit olarak " + yakit + " kullanir");  
 }  
  
 public void vites(String vites) { *// method* System.*out*.println("Araba " + vites + " viteslidir");  
 }

===RUNNER===

public class ParametreliRunner {  
 public static void main(String[] args) {  
  
 ParametreliConstructor oto1 = new ParametreliConstructor();  
 System.*out*.println(oto1.marka + " " + oto1.model + " " + oto1.yil + " " + oto1.kazasiVarMi);  
 *// default degerleri yazdirir* ParametreliConstructor oto2 = new ParametreliConstructor(2015);

*//uygun olduğu için 1. Soruya gider cevabı buradan alır* System.*out*.println(oto2.marka + " " + oto2.model + " " + oto2.yil + " " + oto2.kazasiVarMi);  
 *// null null 2015 false* ParametreliConstructor oto3 = new ParametreliConstructor("Toyota", "Corolla" , 2010 , true);

*//uygun olduğu için 4. Soruya gider cevabı buradan alır* System.*out*.println(oto3.marka + " " + oto3.model + " " + oto3.yil + " " + oto3.kazasiVarMi);

**//Toyota Corolla 2010 true**

public class MyConstructor {  
 int x =3;  
 int y =5;  
  
 MyConstructor () { // 6. Olaraka buraya geldik  
 x+=1; //7…… herhangi bir x le gelmedik, yukarıda verilen x=3 için buradaki sonuç, x=4  
 System.*out*.println("-x"+x); } //-x4…. Şimdi geldiğimiz yere gidiyoruz  
  
 MyConstructor (int i) { //4. Olarak (3) verisiyle buraya geldik.  
 this(); //5. Buraya. Buda bizi parametresiz metoda yollar  
  
 this.y=i; //8…. Burada y yerine i kullan yani yukarıda int i deki i yi,tek veriyle this(3)ten 3 le gelmiştik, o yüzden y=3  
 x+=y; //9….. x=4 tü y=3 bu yüzden işlem sonucu 4+3=7  
 System.*out*.println("-x"+x); } //10. Olarak buradayız çıktımız –x7….geldiğimiz yere gidiyoruz  
  
 MyConstructor (int i, int i2) { //2. Olarak (4,3) verisiyle buraya gelinir  
 this(3); //3. Buraya, buda bizi tek verili metoda yollar  
  
 this.x-=4; //11. Olarak buradayız…x ten 4 çıkar, x=7 idi sonuç x=3  
 System.*out*.println("-x"+x); } //12. Son olarak çıktımız –x3  
 public static void main(String[] args) {  
 MyConstructor mc1 = new MyConstructor (4,3); //main method **1.** Yani başlangıç, iki verili metoda yolluyor

-------------------------------------------------------------------------------------------

public class Static1 {  
 static String *isim*="Mehmet";  
 int yas=49;  
 public static void main(String[] args) { *// static variable'lar olusturulan tum objeler icin gecerlidir (okul adi gibi)  
 // mesela bir class'da 1 instance, 1 de static variable olsun* Static1 st1= new Static1();  
 Static1 st2= new Static1();  
 Static1 st3= new Static1();  
  
 st1.*isim*="Hasan";  
 st1.yas=25;  
 System.*out*.println(st1.*isim* + " " + st1.yas); *// Hasan 25* st2.*isim*="Ayse";  
 st2.yas=30;  
 System.*out*.println(st2.*isim* + " " + st2.yas); *// Ayse 30* System.*out*.println(st1.*isim* + " " + st1.yas); *// Ayse 25(isim static olduğu için her yerdeki metoddaki değişikliği gittiği yere taşıyor, yaş instance old için değişikliğini sadece o classta kullanıyor)* st3.*isim*="Alican";  
 st3.yas=40;  
 System.*out*.println(st3.*isim* + " " + st3.yas); *// Alican 40* System.*out*.println(st2.*isim* + " " + st2.yas); *// Alican 30* System.*out*.println(st1.*isim* + " " + st1.yas); *// Alican 25*

*----*

public class Static2 {  
 int x;  
 static int *y*;  
  
 Static2 (int i) { // 3. Buradayız x e ve y ye i ekle, i=2 le geldik

x+=i; //4. Buradayız.. x=0+2=2 (x default 0) //7. Buradayız. X instance old. İçin 3 cevabı kalıcı olmadı, x=0 ve 3 ekleyince =3   
*y*+=i; //5. Buradayız. y=0+2=2 (y default 0).geldiğimiz yere dönüyoruz..// 8. Buradayız.y static o yüzden y=2 ve 3 ekleyince y =5 }  
 public static void main(String[] args) { // main metod başlangıç 1.  
  
 new Static2(2) ; //2. Buradayız. Bizi “2” ile tek verili metoda yolluyor  
 Static2 dnm = new Static2(3); //6. Buradayız.. bizi tekrar 3 verisiyle yolluyor   
 System.*out*.println(dnm.x + "," +dnm.*y*); //son olarak buradayız ve çıktımız 3,5