

Name:
Id:
Program: (A) (B) Day class Night class

SCORE	Q1	Q2	Q3	Q4	TOTAL

KARABUK UNIVERSITY | COMPUTER ENGINEERING DEPARTMENT
Object Oriented Programming | Midterm | Fall | 11.22.2021 | 17.25 | Duration: 80 mins

QUESTION.1 [21p]: Write outputs in the given table. All the classes are declared in the same package.

Çıktıları verilen tabloya yazın. Tüm sınıflar aynı pakette yazılmıştır.

<pre>enum Type{ ADVENTURE(100,"has an epic journey"), HISTORY(200,"Facts about history"), SCIENCE(300,"Nonfiction"); private final int id; private final String desc; private Type(int id, String desc){ this.id = id; this.desc= desc; } public String getDesc(){ return this.desc;} } public class Book { String name; Type type; String isbn; int year; static int counter; public Book(String bName,Type type){ this(bName,type,2021,++counter); } public Book(String bName,Type type,int y){ this(bName,type,y,++counter); } private Book(String bName,Type type, int y, int count) { this.name= bName; this.year= y; this.isbn= y+"-"+count; this.type= type; } @Override public String toString(){ return "ISBN: "+isbn+"-"+name+"-"+type; } } // End of Book class</pre>	<pre>public class Library { static int index; Book[] books; public Library(int i){ books = new Book[i]; } void addBook(Book b){ books[index] = b; index++; } public void bookData(Book b){ System.out.println("ISBN: "+b.isbn+"-"+ b.name+"-"+b.type.getDesc()); } } // End of Library class public class MainExamClass { public static void main(String[] args) { Book b1= new Book("My Adventure",Type.ADVENTURE); Book b2= new Book("Our History",Types.HISTORY,2010); System.out.println(Book.counter); //(01) System.out.println(b1); //(02) Book b3= new Book("Math",Type.SCIENCE); B2.type=b3.type; Book b4= new Book("my voyage",Type.ADVENTURE,2015); Library ourLibrary = new Library(3); System.out.println(b2.type+"-"+b2.type.getDesc()); //(03) System.out.println(b3); //(04) ourLibrary.addBook(b1); ourLibrary.addBook(b2); System.out.println(Library.index); //(05) ourLibrary.addBook(b3); System.out.println(b2); //(06) ourLibrary.bookData(b2); //(07)</pre>
---	---

(01)		(05)	
(02)		(06)	
(03)		(07)	
(04)			

QUESTION.2 [30p]: Write all classes according to given class diagram on the left.

Car
<ul style="list-style-type: none">-int id-int productionYear-int power+static int counter
<ul style="list-style-type: none">+Car(int id, int productionYear)+void setPower(int power)+int getProductionYear()+int getAge()+void startAndMove()+void stop()+String toString()

Garage
<ul style="list-style-type: none">-Car[] cars
<ul style="list-style-type: none">+Garage(Car[] cars)+void printCarsOlderThan(int age)

MainClass
<ul style="list-style-type: none">+static String[] main(String[] args)

In the main method, create one Garage and 4 Car objects, and print cars that older than **n** years in the garage. (**n**: last digit of your student number) Also, print number of Cars in the garage.

Tüm sınıfları, solda verilen sınıf diyagramına göre yazınız. Main metotta, 4 Car nesnesi ve bir Garage nesnesi oluşturun ve Garage'de **n** yaşından eski arabaları yazdırın. (n: öğrenci numaranızın son hanesi) Ayrıca garajdaki Arabaların numarasını da yazdırın.

QUESTION.3 [24pt]: Write necessary lines of code to perform followings

- a. Create a class called **EncapsulatedNFT**, which applies encapsulation (data hiding) to given **NFT** class.

```
public class NFT {  
    public double balance; // balance cannot be less than 0 or greater than 10000  
    public String owner;  
}
```

- b. **EncapsulatedNFT** class should throw an exception (with an error message) **if any illegal argument** (balance) is set.
c. Inside the MainClass, create an instance/object from EncapsulatedNFT, set attributes for it, handle any exception, and print the exception message on the screen.

Soru.3

- a. **NFT** classına encapsulation (kapsülleme, veri gizleme) prensibini uygulayan **EncapsulatedNFT** adlı bir class oluşturun.
b. Illegal bir arguman (balance) girilirse, **EncapsulatedNFT** sınıfı bir **exception** (bir hata mesajıyla) fırlatmalıdır.
c. MainClass'ta **EncapsulatedNFT** sınıfından bir nesne (instance) oluşturun ve degiskenleri atayın, herhangi bir exception durumuyla başa çıkın (handle edin) ve istisna mesajını ekrana yazdırın.

QUESTION.4 [20p]: Find 10 errors in the following Java code. The errors are both compiler and logical errors. For each error, specify the line number and briefly explain how to fix it. Aşağıdaki Java kodunda 10 hata bulun. Hatalar hem derleyici hem de mantıksal hatalardır. Her hata için satır numarasını belirtin ve nasıl düzeltileceğini kısaca açıklayın.

```
1. public class Patient{
2.     private int id;
3.     static int counter;
4.     private String name;
5.     private float temperature=36.5;
6.     public static final String doctor="Jone";
7.
8.     public patient(String name, double temp,String doctor){
9.         this(name,temp,++counter,doctor);
10.    }
11.    private Patient(String name, float temp,int pId, String doctor){
12.        this.name=name;
13.        this.temperature= temp;
14.        this.id = pId;
15.        this.doctor= doctor;
16.    }
17.    public String getName(){
18.        return this.name;
19.    }
20.    public getId(){
21.        return this.id;
22.    }
23.    @Override
24.    public String toString(){
25.        return this.id+": name:"+this.name+"; temperature"+this.temperature;
26.    }
27. }
28. public class Hospital {
29.     static void main(String[] args) {
30.
31.         Patient patient1= new Patient("Ahmed", 37.2f, "Jak");
32.         System.out.println(patient1);
33.         Patient patient2= new Patient("Omer", 37.2f, "Jone");
34.         System.out.println(patient2.getName());
35.         patient2.name=patient1.name;
36.         Patient patient2= new Patient("Ali", 37.2f, "Jan");
37.         System.out.println(patient2);
38.         System.out.println("number of patients:"Patient.counter);
39.     }
40. }
```