INDIAN INSTITUTE OF TECHNOLOGY KHARAGPUR

End-Autumn Semester Examination 2022-23

Date of Examination: **Session: (FN/AN)** **Duration: 3 hrs.** **Full Marks:100**

Subject No.: CS60059 **Subject:** **Object-Oriented Systems**

Department/Center/School: **Computer Science and Engineering**

Specific charts, graph paper, log book etc., required NIL

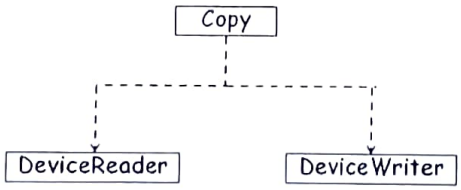
Instructions: Answer all questions

- No clarifications to any question shall be provided. In case you have any doubts, you can make suitable assumptions, but please write down your assumptions clearly.
- All answers should be brief and concise. Lengthy, ambiguous, and irrelevant answers will be penalized.

---

1. Suppose the design of an application software violates the "Law of Demeter" GRASP pattern. Identify an important problem that might be faced during maintenance of the application software. Briefly justify your answer using at most two sentences. **[2+1]**

2. The following class diagram was designed by a programmer to copy information from one device to another. The **Copy** class initializes the reader and writer devices and invokes the **read()** method on the **DeviceReader**. Then it writes the read data to **DeviceWriter** by invoking its **write()** method after making necessary format changes. Name the design principle that it violates and give an improved design. **[1+3]**



3. Consider a Library Automation application to automate the book keeping activities of a Library. The **Library** has a large number of **Issuables**. Each **Issuable** is either a **book**, a **DVD**, or a **Music CD**. The state of each **Issuable** can either be **on shelf**, **issued out**, or **on reserve**. The response to an **issueOut()** request to an **Issuable** object depends on the state of the object. Design the class model of the Library Automation Application using state pattern. Restrict your class model to only what is described in the question. **[5]**

4. Consider that you are developing an application program in which you want all messages and warnings to be written to a log file by calling the **LogMessage()** method of the **LogManger** class. You don't want accidental creation of multiple **LogManger** objects, nor do you want to get into the trouble that may arise when two different messages are attempted for logging in at the same time. Write skeletal Java code for the **LogManager** class, given that it implements the Singleton pattern. **[5]**

5. An email contains a mail header and a mail body. The mail body contains text, image, video and various types of attachments. An attachment, in turn may contain text, image, audio, video, and other types as attachments. It should be possible to display the entire content of an email by calling its **display**() method. Design a class diagram for an email using an applicable design pattern. **[5]**

6. Draw UML class diagram corresponding to the following Java code and identify the design pattern it encodes. **[4+1]**

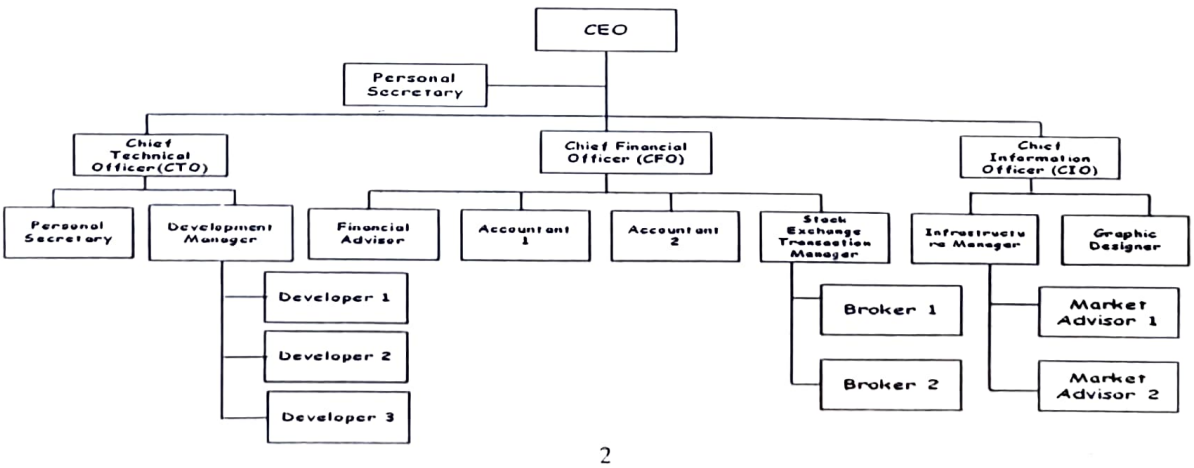
```
class Students implements Persons{
  private OurList namelist = new OurArrayList();
  private OurList addresslist = new OurLinkedList();
  void addName(String name){
     namelist.add(name);}
  int count(){
     return namelist.count();}
  void addAddress(String address){
     addresslist.add(address);}
}
```

```
class OurLinkedList implements OurList{
  private LinkedList <String> addresslist = new
                                  LinkedList<String>();
  int count(){
     return namelist.size();}
  void addAddress(String address){
     addresslist.add(address);}
}
```

```
interface OurList{
   void add(String str);
   int count();
}
```

```
interface Persons{
   void addName(String str);
   int count();
Void addAddress(String str);
}
```

7. Consider that an organization has asked you to implement a software that would help it to automate its employee management activities such as leave approval, and enabling and disabling various employee privileges as promotions and reorganizations take place. All employee management activities should be based on a hierarchical reporting structure as shown in the following diagram. Besides the top level managers: CEO, CTO, CFO, and CIO, the managerial cadre also includes technical managers such as Development manager, Stock Exchange manager, and Infrastructure manager. A technical manager is an employee who leads some engineers, technicians, and support personnel, all of whom are non-managerial employees, and also some junior technical managers. A top level manager usually leads some technical managers and some employees. Which design pattern would you choose to use to implement the corporate hierarchy? Design an appropriate class diagram. **[1+5]**



8. Suppose that a set of sensors are to be placed in the IIT(Kgp) weather monitoring station to sample various weather parameters such as temperature, pressure, humidity, precipitation, and wind speed. A weather monitoring application is required to be developed in Java. It should run on a desktop placed inside the weather monitoring station. Whenever a significant (1%) variation over the last reported reading occurs, the corresponding sensor would invoke the **report()** method of the **WeatherMonitor** class. The real-time weather data would be displayed at different locations in the campus. A large display terminal at the entrance to the academic campus is planned to display all the current weather parameters in a tabular format. A display terminal in front of the administrative building would display the weather parameters over the last year's data in the form of a set of graphs. The Institute website would report the daily maximum and minimum temperature, total rainfall, and average humidity. It is anticipated that in future weather data would be displayed at many more places in the campus, possibly using innovative formats. Please note that at any time, not only should the displayed weather data be consistent among themselves, but should be consistent weather monitor data as well. Design a class diagram for the weather monitoring application using applicable design pattern. Name the pattern you are using. **[5+1]**

9. Assume that you are employed as a programmer by a company manufacturing embedded systems. You are entrusted to develop an application that would help automate sales of various embedded systems of the company. In your sales application, you would have to compute the cost of any embedded system from the costs of the basic components used in the system. Every embedded system is constructed from a few basic components such as CPUs, buses, CD drives, memory, sensors, actuators, and also a few other subsystems. Each such subsystem may in turn contain CPUs, buses, CD drives, memory, sensors, actuators, and possibly a few other subsystems. As part of developing the sales application for embedded systems, design a class diagram to model any arbitrary embedded system that your company manufactures. Name the design pattern that you are using. **[5+1]**

10. Assume that you have set up a startup company named **GreenHOST**, which is a is a web hosting company. Your company offers to the clients three different basic server plans: personal, professional, premium. Each basic plan has certain cost of subscription per month. In addition to subscribing to one of the basic plans, a customer has the choice to subscribe one or more of the various optional hosting services available. The hosting services that are available now are: Cloudhosting, SSL,24x7 support, backup, Google Adwords, CloudFlare, and mySQL. Design a class structure for the customer subscription so that by using your application, the total monthly service fees payable by a customer for his chosen services can be easily computed (an example output is shown below). Note that customers should be able to subscribe additional hosting services or drop any already subscribed services they do not need any more. Further, your application must be able to easily support adding new types of hosting services that may become available in future. Give the class diagram for your solution. Name the design pattern you have used. **[5+1]**

Example output of a customer's service charge:

Customer abc: Subscribed >> Personal Hosting, SSL, Google Adwords, MySQL: Rs. 5994 per month.

11. Consider a banking application in which **Account** class supports a set of methods such as **deposit()**, **withdraw()**, **createDeposit()**, **breakDeposit()**, etc. These methods behave differently depending on the type of the Account. The state of an account can be any combination of its: i) **Account status**: either Dormant or Active, ii) **Account holder status**: either Senior citizen account or normal account, iii) **Account type**: either savings account or current account. A new account can be opened in the state [active, normal, savings] or any such possible combinations. An active account becomes dormant, if it is inoperative for a year. A dormant account can be made into an active account by the manager. A normal account is automatically converted into senior citizen account when the age of the account holder exceeds 60 years. The account holder can fill in the appropriate forms, based on which a savings account can be made into a current account or vice versa by the branch manager.

   a. Draw state machine model of an account object. **[4]**
   b. Design Account class using state pattern. **[5]**

12. Consider the following Java code. Re-write the code using Decorator pattern, without changing the overall behaviour of the code and draw the corresponding UML class diagram. **[5+5]**

```
class A {f(){print("A");};}
class B extends A {f(){super.f(); print("B");}}
class C extends A {f(){super.f(); print("C");}}
class BC extends A {f(){super.f(); print("B"); print("C");}}
class CB extends A {f(){super.f(); print("C"); print("B");}}
```

13. Assume that in an educational Institute, a web client can make a request to any web server by using the **WSI** Interface class given below. The Institute wants to block certain **WebServers** (URLs), so that web clients cannot connect to those **WebServers** (URLs). Every **WebServer** class implements the **WSI** interface class. As and when the Institute administration gets any complaints about a **WebServer,** they would add it to the blocked list. In this situation, which design pattern will be useful to block access to a set of URLs? Give your class design, and write skeletal Java code to implement your solution. **[1+4+5]**

```
public interface WSI {
    void makeRequest(String url);
}
```

14. An arithmetic expression **E** can be an integer or an expression of the form **E op E**. Where **op** is a binary arithmetic operator such as +, -, *, and /. A few examples of valid expressions are **1, 1 + 2, 3 + 4 * 5**. An expression **E** should support a method **eval()** that would display the number obtained by evaluating the expression.

   a) Draw a class diagram to represent an expression **E**. Use applicable design pattern. **[8]**
   b) Draw object diagram for the expression **6 + 3 * (2+1)** **[2]**

15. For each of the following purposes described below, name the most appropriate design pattern that is applicable. In each case, write at most one brief sentence to justify your choice. **[1×10=10]**

a) Add functionality to individual objects dynamically and transparently.

b) Expose the functionality of an object through a different interface.

c) Keep a count on the number of clients accessing a server object, and destruct the server object when no clients are accessing it.

d) Load an object from a database on demand.

e) An instructor wants to schedule an exam for which he/she wants that a notification be sent to all the registered students automatically.

f) As new responsibilities for an object get defined during development and maintenance, prevent the base class from getting subclassed along orthogonal hierarchies.

g) You want to wrap a library class that manages data connections so that it logs all connection attempts as they occur.

h) Provide different levels of access privileges to an object to different clients.

i) Allow a client to use one of several implementations of an interface that can be decided upon dynamically.

j) Realize a read-only collection of "Student-semester-performance" objects.

--------- **The End** -------