Department of Computer & Systems Engineering
January, 2019

ير 2019

Course title Number: Programming -2 (22.1)

م المقرر والرقم الكودي له برمجة-2 (٢٢.١)

2nd Year

نة الدراسية: الثانية

Time allowed: 3 Hours

من: 3 ساعات

Read these requirements carefully:

It is required to build our own version of a *password depot* web application. A password depot application lets the users store all their sensitive data (typically passwords) in one secure place. In order to do so, the user must create *one* account using a username and password at the password depot web site. From this moment on, this is only password the user has to remember. After login, the user has access to all his/her sensitive information. The user can create an entity. This entity can be of type:

- website: having URL, username and password
- WiFi: having SSID and password
- Credit Card: having type, name, number, expiry date, and CCV code

The user has a search text box to search for the entity. The system returns the entities that best match the search. The user can select an entity to view it and/or edit the values of its fields. The user can also delete the whole entity.

The user can define other types of entities, such as a *bank account* information. The user would choose the option create master entity type and give it a name. The user adds the fields that must be present for this entity type. For example, a bank account entity type has a bank name, SWIFT code, account number, username and a PIN. The user defines a standard validator for each field, such as bank name is alphabetic with maximum length of 20 characters, the SWIFT code is alphanumeric with length 10, and the PIN is a 6-digit number, and so on. Having done this, the new entity type is available as an option when the user wants to create an entity.

The user has the chance to export all the entities he/she has as a PDF document.

Answer All 30 Questions. Choose between A, B, C, D, and E

1. Drawing the use case diagram, the diagram will have:
   a. 1 person: representing a typical user
   b. 2 persons: 1 representing the typical user and 1 representing the system
   c. 2 persons: 1 representing the typical user and 1 representing the administrator
   d. 3 persons: 1 representing the typical user, 1 representing the system and 1 representing the administrator
   e. 4 persons: 1 representing the typical user, 1 representing the system, 1 representing the exporting user only and 1 representing the administrator

2. Drawing the use case diagram, the diagram will have the following actions:
   a. Login, create entity, search, edit, delete, and create master entity type.
   b. Create account, login, create entity, search, edit, delete, create master entity type and export.
   c. CRUD operations and export
   d. login, create entity, search, edit, delete, and export.
   e. None of the above.

3. The relationship between the entity and its fields is
   a. a dependency relationship because there is a semantic relationship in which a change in the value of the fields affects the semantic of the entity.
   b. an aggregation since the entity *has* fields and fields are considered as *parts* of the entity
   c. a simple association since fields and entities can exist independent of each other. There is only a *many*-to-*many* relationship between entity and field.
   d. a composition since the fields cannot *exist* without the entity and it does not make sense that an entity *exists* without having fields
   e. a simple association since fields and entities can exist independent of each other. There is only a *one*-to-*many* relationship between entity and fields.

4. In the UML diagram of this system,
   a. there is a ternary relationship
   b. there is no ternary relationship
   c. all relationships are binary of type many-to-many
   d. (a) and (c)
   e. (b) and (c)

5. In UML 2.2 standard, the best diagram to describe the behavior of this system is
   a. the sequence diagram
   b. the class diagram
   c. the state diagram
   d. the use case diagram
   e. the profile diagram

6. In order to generate the PDF, the system must visit and format every entity for the specific user using the
   a. Iterator design pattern
   b. Iterator design pattern and interface or abstract class fundamental design pattern
   c. marker interface and the iterator design patterns
   d. immutable design pattern
   e. dynamic reporting design pattern

7. In order to create a new entity, the best design pattern would be
   a. Object pool
   b. Builder
   c. Factory
   d. Cache Management
   e. Adapter

8. The wizard, creating the user-defined entity type, uses the following design pattern to create the definition of the user-defined entity type:
   a. Builder because the wizard uses complicated logic to construct the user-defined entity type
   b. The prototype to prepare for cloning the user-defined entity type later on.
   c. The factory in order to prepare the user-defined entities to be created within the factory.
   d. The object pool since the system will create many user-defined entities later on.
   e. The command to enable the undo of the creation.

9. In order to create a new entity of type user-defined (e.g., bank account), the best design pattern is:
   a. Prototype
   b. Builder
   c. Cache management
   d. Abstract Factory
   e. None of the above

10. In order to guarantee that no one copies the URL from a user browser and use it to hijack the session on another computer, the system installs a cookie at the web browser of the client and couples it to the session id. With each incoming request, the system has an object that intercepts the call and makes sure that the http request is coming from the same web browser. If this is true, the request is served otherwise the system returns an error message and logs out the user. This object (or group of objects) follow(s) the design pattern
   a. Decorator
   b. Visitor
   c. Proxy
   d. Façade
   e. Adapter

11. Creating a new session for a newly logging user is resource intensive, since the database has to be queried to validate the username and password. Also, the system wants to limit the maximum number of concurrent users to 1000. The best way to achieve this target is to manage the session objects using the following design pattern:
   a. Proxy
   b. Observer
   c. Object pool
   d. State
   e. Strategy

12. The best design pattern to implement the configurable data validators for each field is
   a. Builder
   b. State
   c. Command
   d. Strategy
   e. None of the above

13. In general, what is the drawback of the object pool design pattern?
   a. The object pool is slow
   b. The object pool consumes lots of memory
   c. The programmer must manage the release of resources
   d. The programmer must implement a lot of small classes that confuse the reader maintaining the code later on
   e. The object pool has no drawbacks

14. If we decide to have a mobile version of the application in addition to the web-based version, the MVC would make this extension easier because:
   a. MVC is based on Swing, which is available on mobile devices.
   b. the view and controller classes can be reused, only the model classes must be newly implemented.
   c. the model and view classes can be reused, only the controller classes must be newly implemented.
   d. the model and controller classes can be reused, only the view classes must be newly implemented.
   e. MVC will not make things easier because it is hard to debug

15. If the user changes the value of a field in the web browser, the mobile application should show the change at once. This is best implemented using the
   a. Command design pattern
   b. Observer design pattern
   c. Auto-notify design pattern
   d. Producer-consumer design pattern
   e. State design pattern

16. The user is mentally prepared to the fact that the login process would take several seconds. Yet, once navigating inside the system, the user wants to retrieve his/her entities very quickly. The best way to do this is to implement
   a. the cache management design pattern loading the entities during login and keeping them in memory throughout the session.
   b. the object pool to keep the entities in memory after the user logs out for a future login.
   c. the MVC design pattern to speed up the display of entities.
   d. the virtual proxy design pattern to create the illusion that the entities are being displayed whenever the entities are retrieved.
   e. Nothing helps, we just need faster servers.

17. In order to implement the façade design pattern properly, the following should hold:
   a. the number of exposed classes in the package is reduced to the minimum, preferably one
   b. methods in classes other that the façade class should be declared public
   c. methods in classes other that the façade class should be declared private or protected
   d. (a) and (b)
   e. (a) and (c)

18. In the password depot system, the decorator design pattern can be used
   a. to decorate the web site with good fonts and icons
   b. to nicely format the exported PDF file
   c. to show the password field in the form of stars with an "eye" icon to unlock the field
   d. to allow for special letters such as Arabic characters
   e. in none of the above

19. In the decorator design pattern, the following is true:
   a. The methods of the optional wrapper class override the methods in the decorated class using inheritance
   b. The methods of the optional wrapper class override the methods in the decorated class using polymorphism
   c. The methods of the optional wrapper class implement the same interface as the decorated class
   d. There is no relation between the methods of the optional wrapper class and the methods of the decorated class.
   e. None of the above

20. In the cache management design pattern, which object is removed from the cache?
   a. Least recently used
   b. Least frequently used
   c. Most recently used
   d. Configurable (a) or (b)
   e. Configurable (b) or (c)

21. In the chain of responsibility design pattern,
   a. the sender and receiver have no explicit knowledge of each other
   b. receipt is not guaranteed - a request could fall off the end of the chain without being handled
   c. the chain of handlers can be modified dynamically
   d. all the above is true
   e. some but not all of the above hold

22. The notify() method in the observer design pattern
   a. is overridden in the concrete subjects
   b. calls the update() methods in all observers
   c. is called by all update() methods in all observers
   d. must run in a separate thread
   e. must have a wait() method in be called in each observer

23. We use the adapter design pattern in the case of:
   a. Adapting a new system to the logic of an old system
   b. Adapting to a different interface of another system
   c. Adapting the code to a different coding style
   d. Adapting the observer to the subject
   e. Adapting the order of execution of commands

24. How much info about the change should the subject send to the observers in the observer design pattern?
   a. Push Model – Lots
   b. Push Model – Very Little
   c. Pull Model - Very Little
   d. (a) and (c)
   e. (b) and (c)

25. State one unhappy case in the modelling of the application.
   a. The server gets a hardware failure.
   b. The UI looks messy and bad.
   c. The user gives in a wrong combination of username and password.
   d. The user deletes an entity.
   e. The user requests to unregister him-/herself in the system.

26. Producer consumer design pattern, the buffer can be implemented as
   a. Queue
   b. Stack
   c. Hashmap
   d. (a) or (b)
   e. (a) or (c)

27. The following holds in a mutli-threaded program:
   a. A thread can start on any object that implements the Runnable interface.
   b. The Runnable interface has only single method called "run" which is the start code of the associated thread.
   c. A thread can be either in state: running, or sleeping, or suspended, or stopped, or busy
   d. All the above is true
   e. Some but not all of the above hold

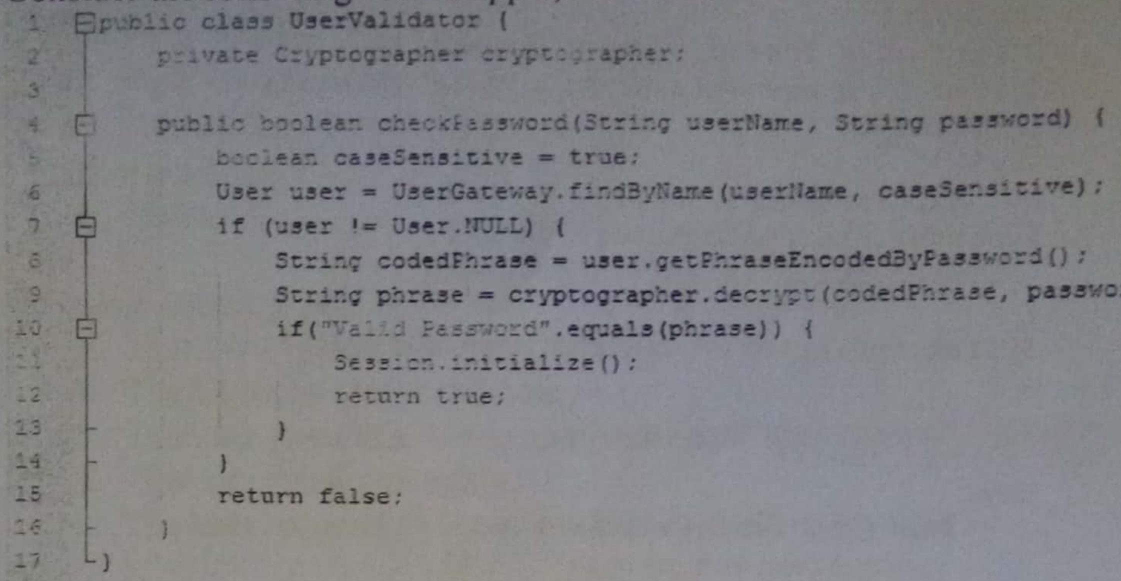
28. Consider the following code snippet,

```
class ThreadDemo implements Runnable{
     ThreadDemo(){
          Thread ct = Thread.currentThread();
          Thread t  = new Thread(this,"Demo Thread");
          t.start();
          try{
              Thread.sleep(2500);
          } catch(InterruptedException e){
            System.out.println("interrupted");
          }
          System.out.print("!exiting main thread!");
     }

     public void run(){
          try{
              for (int i=5; i>0; i--) {
                  System.out.print(" "+ i);
                  Thread.sleep(1000);
              }
          } catch(InterruptedException e){
              System.out.println("interrupted");
          }
          System.out.print("!exiting child thread!");
     }

     public static void main(String args[]){
          new ThreadDemo();
     }
}
```

what is the most likely output?

a. 5 4 3 2 1 !exiting child thread! !existing main thread!
b. 5 4 !existing main thread! 3 2 1 !exiting child thread!
c. 5 4 !existing main thread!
d. 5 4 3 !existing main thread! 2 1 !exiting child thread!
e. 5 4 3 !existing main thread!

29. Consider the following code snippet,



```
public class UserValidator {
    private Cryptographer cryptographer;

    public boolean checkPassword(String userName, String password) {
        boolean caseSensitive = true;
        User user = UserGateway.findByName(userName, caseSensitive);
        if (user != User.NULL) {
            String codedPhrase = user.getPhraseEncodedByPassword();
            String phrase = cryptographer.decrypt(codedPhrase, password);
            if("Valid Password".equals(phrase)) {
                Session.initialize();
                return true;
            }
        }
        return false;
    }
}
```

The side-effect of the method checkPassword is:

a. The method crashes if the user is not found.

b. The session is initialized whereas the responsibility of this method is only to validate the username and password

c. The method changes the password although its responsibility is only to validate the username and password

d. The method does not state why the validation fails

e. The method has no side effects.

30. Identify a design pattern referenced by this code

a. Cache management

b. State

c. Delegation

d. Null object

e. Composite

GOOD LUCK

K.Nagi