Please take few moments to write down your information.

Name: _____ ID: _____

Major: _____

TOTAL MARKS 12, MAXIMUM SCORE 10

(Q1) Propose an architecture style for an e-mail system that filters incoming e-mails with a whitelist (e-mails from senders on the whitelist are accepted), a blacklist (e-mails from senders on the blacklist are deleted), and the SpamAssassin tool (e-mails that do not pass this check are marked as spam). The system will run on a single-core server machine, but may be moved to a multi-core server if the load gets too high – Justify your answer. (2 Points)

Two accepted answers:

(1) Pipes and Filters, Unless filtered out, a typical email passes through the whitelist module, the blacklist module, then ends with the SpamAssassin Module. This is a typical application for the Pips and Filters. (**accepted answer – TWO POINTS**)

(2) Service oriented Architecture Style, each module can be deployed as a service and the system could call services upon need (**accepted answer – Two POINTS**)

Marking Notes:

The style is worth ONE point & the justification is worth ONE Point

The three modules used in sequence, with no UI mandates; hence, Client/Server, Peer-To-Peer and Tiered architecture styles do not fit.

No UI requirements, hence Model/View/Controller does not fit

Each mail is handled independently; hence, Repository does not add any extra value.

(Q2) While designing web-based system, the team architect suggested using session-cookies to implement the access control matrix.

i. **In view of your understanding of approaches to handle global resources, where do session-cookies fit (which approach)? (1 Points)**

Three accepted answers:

Capability approach. In this case, the cookie contains all information about the user access rights. (**ONE POINT**) – accepted answer with or without justification

Global Access Table – the cookie contains only the user ID access rights are handled at the system side. (**ONE POINT**) – accepted only when justification is provided.

Access control list – the cookie contains only the user ID access rights are handled at the system side. (**ONE POINT**) –accepted only when justification is provided.

With no justifications, **capability** is the right answer.

ii. **Do you agree/disagree with this suggestion? Justify your answer (2 Points)**

This is a good approach as web applications are stateless by definition, having the user to maintain his capabilities (or ID) is good for handling multiple users accessing the web server simultaneously And variable user loads. Security issues could be mentioned but are not major to deduct/add marks.

For Marking, keep in mind that the selected method dictates the justification. Other reasonable answers are accepted for (anything from 0.5 to 1.5 points).

(Q3) The following code violates at least one of the SOLID design principles.

```
public static String getAnimalNoise(Animal animal) {
  if (animal instanceof Dog)
    return "Woof";
  if (animal instanceof Cat)
    return "Miau";
  return "";
}
```

Mark the principles that it violate and indicate why. (4 Marks)

SRP (No Violated) (**HALF POINT**)

OCP (Violated) Adding more animals to the list will mandate changing the code. (**ONE POINT**)

LSP (Violated) If another type of animal (say a tiger), the method will get will reply with an empty string. This means that the method is not correct for any subtype of `Animal`. **(ONE POINT)**

ISP (Not violated) **(HALF POINT)**

DIP (2 accepted answers) **–**

(Violated) The "`if`" statement depends on implementation details (`instanceof`) **(ONE POINT)**

(Not violated) accepted answer for less marks since the violation is a side effect of the previous violations and not the core of the problem **(HALF POINT)**

The correct implementation for this method should be the use of polymorphism with a `getNoise` inside the Animal class. (**not part of the answer**)

(Q4) Specify which of the following inheritance relationship is specification inheritance and which is implementation inheritance (1 point each – 3 Points max)

a. **A rectangle class inherits from a polygon class** Specification
b. **A player class inherits from a user class** Specification
c. **A set class inherits from a binary tree class** Implantation
d. **A window class inherits from a polygon class** Implantation.

The Best Of Luck,,,

Amr Kamel