|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **National University of Computer and Emerging Sciences, Lahore Campus** | | | | |
| C:\Users\saif\AppData\Local\Microsoft\Windows\Temporary Internet Files\Content.Word\final design.jpg | **Course:** | **Software Design and Analysis** | **Course Code:** | **CS-3004** |
| **Program:** | **BS (Computer Science)** | **Semester:** | **Fall 2024** |
| **Duration:** | **45 Minutes** | **Total Marks:** | **25** |
| **Quiz Date:** | **30-Oct-24** | **Roll No.** |  |
| **Section:** | **BCS-5B** | **Name:** |  |
|  |  |  |  |
|  |  | | | |

**Question 1) Define Coincidental Cohesion and Functional Cohesion along with examples. (5 Marks)**

Coincidental cohesion is the lowest level of cohesion. It occurs when parts of a module are grouped together arbitrarily, with little or no meaningful relationship between their functionalities. The tasks in a coincidentally cohesive module are unrelated and often don’t contribute to a single, well-defined purpose.

Example: Consider a module named Utilities that includes functions for:

* Printing a report,
* Sending an email,
* Calculating discounts,
* Logging errors, and
* Performing basic math operations.

These functions don’t serve a common purpose; they’re arbitrarily grouped together, making this module coincidentally cohesive.

Functional cohesion is the highest level of cohesion. It occurs when all elements in a module work together to achieve a single, well-defined task or purpose. Each function or process within the module contributes directly to its main responsibility.

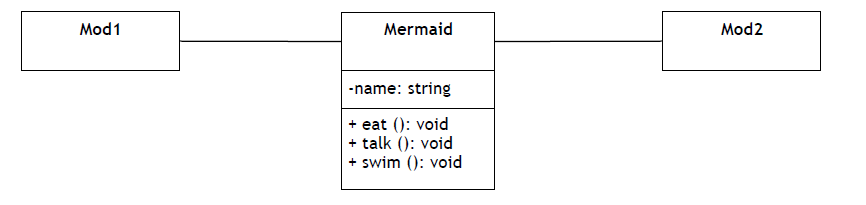
Example: Consider a module named OrderProcessing that performs tasks related to processing an order:

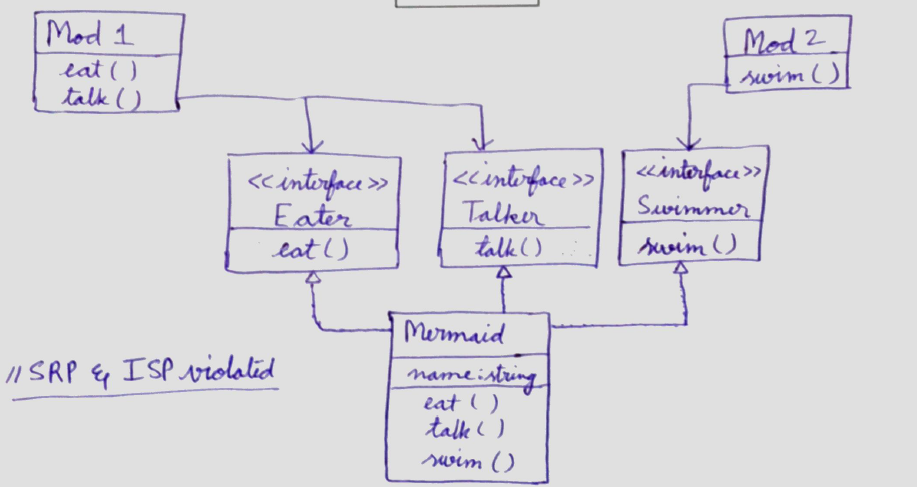
* Validating the order,
* Checking inventory,
* Calculating the total price,
* Applying discounts, and
* Confirming the order.

Each of these tasks contributes to the overall goal of processing an order, making this module functionally cohesive.

**Question 2) (10 marks)**

Consider a video game that has different creatures and characters. One interesting creature is a mermaid. The mermaid eats and talks like a woman, and swims like a fish. The Mermaid class provides all these three functions. This game has several modules. One of these modules (Mod1) uses only eat and talk functions of the Mermaid class while another (Mod2) uses only the swim function. Refactor (improve) the design given below in the light of SOLID principles. Also Mention the principles that are violated by this design.



****

**Question 3) (10 marks)**

A cricket-based mobile game – FASTCric – is designed in such a way that after a bowler delivers a ball to a batter, the batter’s play function is called. This play function has two parameters i.e. ball type (leg spin, off spin, yorker, or bouncer) and ball speed (km/hour). This function decides which type of shot is played by the batter using the following rules:

* if leg spin or off spin is bowled, the batter plays the sweep shot
* if yorker is bowled, the batter plays the block shot
* if bouncer is bowled and ball speed is less than 80 km/hour and batter’s energy level is at least 70%, then hook shot is played
* in all other cases, the leave-it-alone shot is played.

All types of shots keep track of the runs scored when they were played and the play function also returns the runs scored. A sweep shot always result in 4 runs while a hook shot always results in six runs. Both block shot and leave-it-alone shot result in 0 runs.

**Model a UML 2 design class diagram depicting a portion of the design of FASTCric**

**Important Instructions: This diagram should have exactly 7 classes and exactly 1 enumeration.**

A computer screen shot of a computer screen

Description automatically generated