Risheet Aggarwal Mehta(2020A7PS1201P)

Utsav Goel(2020A7PS0984P)

Q3.)

ENCAPSULATION:- In object-oriented computer programming languages, the notion of encapsulation (or OOP Encapsulation) refers to the bundling of data, along with the methods that operate on that data, into a single unit. Many programming languages use ***encapsulation*** frequently in the form of ***classes***.

For example—In our class Monopoly we both data objects of die and board and at the same time we have a method startGame function.

COMPOSITION OVER INHERITENCE:- In object-oriented programming, the composition is the architecture strategy for executing a relationship between objects. Java composition is done using instance variables from other objects.

For example—The function doAction makes an object of classes Player and Board. As this is can be done and we can have the class Chance inherit both Player and Board.

PROGRAM TO AN INTERFACE AND NOT IMPLEMENTATION:-

Coding against interface means, the client code always holds an Interface object which is supplied by a factory.

This has not been used in our program but we could have done this for the class square. So instead of square being a class it would have been a interface.

STRIVE FOR LOOSE COUPLING:-

- It reduces the risk that a change made within one element might create an unanticipated impact on the other elements

- It simplifies testing, maintenance, and troubleshooting ...

The coupling used in our code is quite intensive and any change in one might cause disturbance in some other class. For example if I was to make a change in class square it would cause a disturbance in the classes which inherit Square class.

OBSERVER DESIGN PATTERN

This design pattern is used in the game .

Observer pattern is used when there is one-to-many relationship between objects such as if one object is modified, its depenedent objects are to be notified automatically. Observer pattern falls under behavioral pattern category.