Game Learn to create a class diagram for the Blackjack game using the bottom-up approach. We'll cover the following

Class Diagram for the Online Blackjack

 Components of Blackjack Card

Deck

• Blackjack game view • Blackjack game Enumerations and custom data types

Shoe Hand Players Blackjack controller

 Association Aggregation Composition Inheritance

Relationship between the classes

· Class diagram for the Blackjack game

 Design pattern Al-powered trainer We'll create the class diagram for the Blackjack game. In the class diagram, we will first design the classes, and then identify the relationship between classes according

In this section, we'll define the classes for the Blackjack game. As mentioned earlier,

Card belongs to a suit and has a face value. The face value of the card is according to

the card number. For example, if we have a number card 5, its face value is also 5. The

Card

+ Card(Suit cardsuit, int cardFaceValue)

The class diagram of the Card class

்டு R3: Blackjack Game

Face value

1 or 11

Equals the card number

10

R3: Every card has points associated with it. The criteria to calculate the face

Deck has 52 cards of four suits, and one suit contains nine number cards (2–10) and

four face cards (King, Queen, Jack, and Ace). The Deck class contains a list of cards

Deck

+ getCard() : Card {list}

The class diagram of the Deck class

∹穴̀· R2: Blackjack Game

Shoe

+ Shoe(int numberOfDecks, Deck decks)

The class diagram of the Shoe class

∵;; R1: Blackjack Game

R1: The Blackjack game contains the shoe of cards which contains one or more

The Hand class represents a Blackjack hand used in this game which can contain

Hand

+ Hand(Card card1, Card card2)

The class diagram of the Hand class

்் R8, R9, and R10: Blackjack Gam

R8: The player can draw the additional card if their hand has less than 21 points.

R10: If a player or the dealer's hand is more than 21, they bust and lose the game.

Player is an abstract class. There are two types of players: BlackjackPlayer and

BlackjackPlayer: They place the first wager and update the stake with winning and

Dealer: They are primarily in charge of dealing cards and following the Blackjack

<<Abstract>> Player

+ addHand(Hand hand) : void + removeHand(Hand hand): void + resetPassword(): bool + addToHand(Hand hand)

The class diagram of Player and its derived classes

∹穴̀· R4: Blackjack Game

R4: There can be two types of users that can play the Blackjack game, i.e., dealer

The BlackjackController class validates the action(hit, stand) and responds

BlackJackController

+ validateAction() : bool

The class diagram of the BlackjackController class

The BlackjackGameView class represents the game view. The BlackjackGame class

BlackJackGameView

+ playAction(string action, Hand hand) : void

The class diagram of the BlackjackGameView class

The BlackjackGame class represents how we can play this game or its basic sequence

of play. It controls the game, accepts wagers from players, and distributes cards from

BlackJackGame

+ BlackJackGame(BlackJackPlayer player, BlackJackPlayer dealer)

The class diagram of BlackjackGame class

∵౧ౕ R1: Blackjack Game

R1: The Blackjack game contains the shoe of cards which contains one or more

The following provides an overview of the enumerations and custom data types used

Suit: We need to create an enumeration to keep track of the suit of the card,

status, whether it is active, canceled, closed, blocked, or none.

<<enumeration>>

Suit

AccountStatus: We need to create an enumeration to keep track of the account

Active

Closed Canceled

None

Enums in the Blackjack game

Person

The class diagram of the Person class

Now, we'll discuss the relationships between the classes we have defined above in

The BlackjackGame class has a one-way association with BlackjackGameView.

The Player class has a one-way association with BlackjackGame and

The BlackjackController class has a one-way association with BlackjackGame.

Shoe

BlackJackGameView

<<Abstract>>

Palyer

Hand

BlackjackPlayer

The association relationship between classes

Card

BlackjackGame

The aggregation relationship between classes

The composition relationship between classes

Both the BlackjackPlayer and Dealer classes extend the Player class.

player : BlackJackPlayer

maxNumberOfDacks : int

BlackJackPlayer

+ BlackJackPlayer(Hand hand

The class diagram of the Blackjack game

The Iterator design pattern can be applied, since cards are assigned to the players

We can also use the State design pattern for our online Blackjack game because this

At this stage, everything should be clear. If you encounter any confusion or ambiguity,

feel free to utilize the interactive AI-enabled widget below to seek clarification. This

tool is designed to assist you in strengthening your understanding of the concepts.

from the deck of cards by just iterating through the list of cards.

game has a finite number of states. Some of these states are as follows:

+ placeBet(int bet) : void

+ start(): void

bet : int

Note: We have already discussed the inheritance relationship between classes

BlackJackGame

+ BlackJackGame(BlackJackPlayer player, BlackJackPlayer dealer) + playAction(string action, Hand hand) : void + hit(Hand hand) : void + stand(Hand hand) : void

BlackJackGameView

+ playAction(string action, Hand hand) : void

manipulates

Dealer

hand : Hand

getTotalScore()

BlackJackController

validateAction(): boo

BlackjackGame

<<Abstract>>

Player

- name : string streetAddress : string

- city: string - state : string - zipcode : int - country : string

Relationship between the classes

The class diagram has the following association relationships:

The Shoe class has a one-way association with Deck.

Deck

BlackJackGame

BlackJackController

The class diagram has the following aggregation relationships.

• Both the Deck and Hand classes contain the Card class.

The class diagram has the following composition relationships.

The BlackjackGame class is composed of Shoe.

Shoe

Hand

The following classes show an inheritance relationship:

in the component section above one by one.

Class diagram for the Blackjack game

Here's the complete class diagram for our Blackjack game:

decks : Deck {list}

numberOfDecks

+ createShoe(): void

+ shuffle() : void + dealCard : Card

+ Shoe(int numberOfDecks, Deck decks)

id : string - password : string

balance : double

status : AccountStatus person : Person

+ resetPassword() : bool + addToHand(Hand hand)

Draw a card and give it to the dealer

Draw a card and give it to the player

addHand(Hand hand): void

removeHand(Hand hand): void

<<Abstract>>

Player

The Player class is composed of Hand.

• The BlackjackGame class contains the Dealer and BlackjackPlayer.

Person: Used to store information related to a person like a name, street

Blacklisted

<<enumeration>>

AccountStatus

the Shoe to hands, updates the game's status, gathers lost wagers, pays winning

wagers, divides hands, and responds to player decisions to hit or stand.

+ playAction(string action, Hand hand): void

- player : BlackJackPlayer

- maxNumberOfDecks : int

+ hit(Hand hand): void + stand(Hand hand) : void

Enumerations and custom data types

whether it is diamond, spade, heart, or club.

Heart Spade

Club

address, country, etc.

our Blackjack game.

BlackjackController.

Association

Aggregation

Deck

Dealer

Composition

Inheritance

cards : Card {list}

faceValue : int

cards : Card {list}

getScore() : int addCard(Card card)

+ Hand(Card card1, Card card2)

Design pattern

Shuffle the deck

Deal cards

Player hit

Player stand

Al-powered trainer

+ Deck() + getCard() : Card {list}

Card

+ Card(Suit cardsuit, int cardFaceValue)

Diamond

- dealer : Dealer - shoe : Shoe

+ start(): void

extends

- hand : Hand

+ getTotalScore()

Dealer

- id : string - password : string balance : double - status : AccountStatus - person : Person - hands : Hand

BlackJackPlayer

+ BlackJackPlayer(Hand hand) + placeBet(int bet) : void

bet:int - totalCash : int

and the player.

accordingly.

Blackjack controller

Blackjack game view

Blackjack game

decks of cards in it.

in this problem:

updates the BlackjackGameView class.

Dealer. These classes can be derived from the Player class.

losing sums. They can choose between hit and stand options.

R9: The dealer can draw an additional card if their hand is less than 17.

- cards : Card {list}

+ getScore(): int + addCard(Card card)

R2: The deck will consist of 52 cards in four suits, where each suit contains 13

cards: Ace, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, Jack, Queen, and King.

Shoe is a device to hold multiple Deck, and it has a shuffle operation.

- decks : Deck {list} - numberOfDecks : int

+ createShoe(): void + shuffle(): void + dealCard(): Card

cards : Card {list}

+ Deck()

- suit : Suit - faceValue : int

face value for the King, Queen, and Jack is 10, and 1 or 11 for the Ace, depending on

we are following the bottom-up approach to designing a class diagram for the

to the requirements for the Blackjack game problem.

Components of Blackjack

Blackjack game.

the situation.

Deck

Shoe

decks of cards in it.

Hand

multiple cards.

Players

rules.

value of the card is as follows:

Card

Ace

From 2 to 10

Face cards (King, Queen, Jack)

where the top card is in the first index.

Card