Method		Oleve Investor Con	****		
	# Test Description	Class: InventorySys Sample Input Data	Expected Output	Actual Output	P/F
ın	The main application loop starts and displays the main menu.	The run() method is called.	The main menu is displayed to the console	The main menu is displayed to the console.	P
un	The main approach toop state and displays the main mena.	Class: Card	The main ment to displayed to the condoic.	The main ment to displayed to the console.	· · ·
Method	# Test Description	Sample Input Data	Expected Output	Actual Output	P/F
getName	1 Returns the correct name of the Card.	Card created with name "Pikachu"	Pikachu	Pikachu	P
getBaseValue	Returns the correct base value of the Card.	Card created with base value 25.50	25		25.5 P
getBasevalue	Returns the correct base value of the card. Returns the correct Rarity enum constant.	Card created with base value 23.30 Card created with Rarity.RARE	Rarity.RARE	Rarity.RARE	20.0 F
getVariant	Returns the correct Variant enum constant.	Card created with Variant.FULL_ART	Variant.FULL_ART	Variant.FULL_ART	, D
gotvanant	Returns correct value for a normal variant.	Base value: 50.0, Variant: Variant.NORMAL	5		50 P
	2 Returns correct value for Extended-art variant.	Base value: 50.0, Variant: Variant.EXTENDED_ART	7		75 P
	3 Returns correct value for Full-art variant.	Base value: 50.0, Variant: Variant.FULL_ART	10		100 P
getCalculatedValue	4 Returns correct value for Alt-art variant.	Base value: 50.0, Variant: Variant.ALT_ART	15		150 P
gerouiouidea vaide	Treating context raide for the diff variant.	Enum: Rarity		1	100 1
Method	# Test Description	Sample Input Data	Expected Output	Actual Output	P/F
mounou	1 Returns correct display name for RARE.	Rarity.RARE.getDisplayName()	Rare	Rare	Р.
getDisplayName	2 Returns correct display name for COMMON.	Rarity.COMMON.getDisplayName()	Common	Common	Р.
g	1 Returns correct enum for valid input 1.	Rarity.fromInt(1)	Rarity.COMMON	Rarity COMMON	P
	2 Returns correct enum for valid input 4.	Rarity.fromInt(4)	Rarity.LEGENDARY	Rarity.LEGENDARY	P
	3 Returns null for invalid zero input.	Rarity.fromInt(0)	null	null	P
fromInt	4 Returns null for invalid out-of-bounds input.	Rarity.fromInt(5)	null	null	P
		Enum: Variant			
Method	# Test Description	Sample Input Data	Expected Output	Actual Output	P/F
getDisplayName	Returns correct display name for FULL_ART.	Variant.FULL_ART.getDisplayName()	Full-art	Full-art	P P
gene apaymana	Returns correct multiplier for EXTENDED_ART.	Variant.FXTENDED_ART.getMultiplier()	1	5	1.5 P
getMultiplier	2 Returns correct multiplier for NORMAL.	Variant.NORMAL.getMultiplier()	†		1 P
J	1 Returns correct enum for valid input 1.	Variant.fromInt(1)	Variant.NORMAL	Variant.NORMAL	P
	2 Returns correct enum for valid input 1.	Variant.fromInt(1)	Variant.FULL ART	Variant.FULL_ART	P
	3 Returns null for invalid negative input.	Variant.fromInt(-1)	null	null	P
fromInt	4 Returns null for invalid out-of-bounds input.	Variant.fromInt(5)	null	null	P
	The second secon	Class: Binder			i.
Method	# Test Description	Sample Input Data	Expected Output	Actual Output	P/F
getName	1 Returns the correct name of the Binder.	Binder created with name "Trade Binder"	Trade Binder	Trade Binder	Р.
getCards	Returns a defensive copy of the cards list.	Binder with 2 cards. getCards() is called.	New ArrayList with 2 cards	New ArrayList with 2 cards	P
getourus	1 Returns 0 for a new binder	New binder created	TOWN MAY LOCK WAIT L COLORS	n now / may 2 to a not	0 P
getCardCount	2 Returns correct count after adding cards.	5 cards added to binder.		5	5 P
getodiucount	1 Returns false when binder is not full.	Binder with 19 cards.	FALSE	FALSE	P
isFull	2 Returns true when binder is full.	Binder with 20 cards.	TRUE	TRUE	P
ior dir	1 Adds a card to a non-full binder.	Add card to binder with 10 cards.	TRUE	TRUE	P
addCard	2 Fails to add a card to a full binder.	Add card to binder with 20 cards.	FALSE	FALSE	P
dudodiru	Fails to remove card at negative index.	removeCard(-1)	null	null	P
	2 Removes an existing card successfully.	removeCard(0) on a binder with 1 card.	The removed Card object	The removed Card object	Р
removeCard	3 Fails to remove card at out-of-bounds index.	removeCard(5) on a binder with 5 cards.	null	null	
Temoveculu	o i and to territore dark at out or bounds made.	Class: Deck	TMII	THE STATE OF THE S	
Method	# Test Description	Sample Input Data	Expected Output	Actual Output	P/F
getName	Returns the correct name of the Deck.	Deck created with name "My Deck"	My Deck	My Deck	P
getCards	1 Returns a defensive copy of the cards list.	Deck with 3 cards. getCards() is called.	New ArrayList with 3 cards	New ArrayList with 3 cards	Р
getCardCount	Returns correct count after adding cards.	7 cards added to deck.		7	7 P
J	1 Returns false when deck is not full.	Deck with 9 cards.	FALSE	FALSE	Р
isFull	2 Returns true when deck is full.	Deck with 10 cards.	TRUE	TRUE	P
	Returns true if card with same name exists.	Deck has "Pikachu". containsCard("Pikachu")	TRUE	TRUE	P
	2 Returns true for case-insensitive check.	Deck has "Pikachu", containsCard("pikachu")	TRUE	TRUE	
containsCard					P
	3 Returns false if card does not exist	Deck has "Pikachu" containsCard("Charmander")	FALSE		P
	Returns false if card does not exist. 1 Adds a unique card to a non-full deck	Deck has "Pikachu". contains Card("Charmander") Add "Charmander" to deck with 5 cards	FALSE TRUE	FALSE TRUE	P P
	1 Adds a unique card to a non-full deck.	Add "Charmander" to deck with 5 cards.		FALSE TRUE	P P P
			TRUE	FALSE	P P P P
addCard	Adds a unique card to a non-full deck. Fails to add a card if deck is full. Fails to add a card with a duplicate name.	Add 'Chammander' to deck with 5 cards. Add 'Squirtle' to deck with 10 cards. Add 'Rikachu' to deck that already has a "Pikachu".	TRUE FALSE	FALSE TRUE FALSE	P P P P
	Adds a unique card to a non-full deck. Fails to add a card if deck is full.	Add "Charmander" to deck with 5 cards. Add "Squirtle" to deck with 10 cards.	TRUE FALSE FALSE	FALSE TRUE FALSE FALSE	P P P P P
addCard	1 Adds a unique card to a non-full deck. 2 Fails to add a card if deck is full. 3 Fails to add a card with a duplicate name. 1 Removes an existing card successfully.	Add "Charmander" to deck with 5 cards. Add "Squiffe" to deck with 10 cards. Add "Squiffe" to deck with a thready has a "Pikachu". removeCard(2) on a deck with 3 cards. removeCard(1) on a deck with 10 cards.	TRUE FALSE FALSE The removed Card object null	FALSE TRUE FALSE FALSE FALSE The removed Card object	P P P P P
addCard removeCard	1 Adds a unique card to a non-full deck. 2 Fails to add a card if deck is full. 3 Fails to add a card with a duplicate name. 1 Removes an existing card successfully.	Add 'Charmander' to deck with 5 cards. Add "Squirtle" to deck with 10 cards. Add "Pikachu" to deck that already has a "Pikachu". removeCard(2) on a deck with 3 cards.	TRUE FALSE FALSE The removed Card object null	FALSE TRUE FALSE FALSE FALSE The removed Card object	P P P P P P P
addCard	Adds a unique card to a non-full deck. Falls to add a card if deck is full. Falls to add a card with a duplicate name. Removes an existing card successfully. Falls to remove card at out-of-bounds index.	Add "Charmander" to deck with 5 cards. Add "Squirlie" to deck with 0 cards. Add "Skutrie" to deck with 0 cards. Add "Skutrie" to deck that already has a "Pikachu". removeCard(2) on a deck with 3 cards. removeCard(10) on a deck with 10 cards. Class: CollectionMan	TRUE FALSE FALSE The removed Card object null lager	FALSE TRUE FALSE FALSE FALSE In removed Card object	P P P P P P P P P P P P P P P P P P P
addCard removeCard	1 Adds a unique card to a non-full deck. 2 Fails to add a card if deck is full. 3 Fails to add a card with a duplicate name. 1 Removes an existing card successfully. 2 Fails to remove card at out-of-bounds index.	Add "Charmander" to deck with 5 cards. Add "Squritle" to deck with 10 cards. Add "Squritle" to deck that laredy has a "Pikachu". removeCard(2) on a deck with 3 cards. removeCard(10) on a deck with 10 cards. Class: CollectionMan Sample Input Data	TRUE FALSE FALSE The removed Card object null ager Expected Output	FALSE TRUE FALSE FALSE The removed Card object null Actual Output	P P P P P P P P P P P P P P P P P P P
addCard removeCard Method	Adds a unique card to a non-full deck. Falst to add a card if deck is full. Falst to add a card with a duplicate name. Removes an existing card successfully. Falst to remove card at out-of-bounds index. Test Description Tiest Description Finds an existing card by name.	Add "Charmander" to deck with 5 cards. Add "Squirlie" to deck with 0 cards. Add "Squirlie" to deck with 1 cards. Add "Sketchu" to deck that already has a "Pikachu". removeCard(2) on a deck with 1 cards. removeCard(10) on a deck with 10 cards. Class: CollectionMar Sample Input Data findCard("Charizard") when it does not exist.	TRUE FALSE FALSE The removed Card object	FALSE TRUE FALSE FALSE The removed Card object ruil Actual Output Card object for "Charizard" null	P P P P P P P P P P P P P P P P P P P
addCard removeCard Method	1 Adds a unique card to a non-full deck. 2 Falls to add a card if deck is full. 3 Falls to add a card with a duplicate name. 1 Removes an existing card successfully. 2 Falls to remove card at out-of-bounds index. # Test Description 1 Finds an existing card by name. 2 Returns null for a non-existent card.	Add "Charmander" to deck with 5 cards. Add "Squirle" to deck with 10 cards. Add "Squirle" to deck that already has a "Pikachu". removeCard(2) on a deck with 3 cards. removeCard(10) on a deck with 10 cards. Class: CollectionMan Sample Input Data finoCard("Charizard") when it exists.	TRUE FALSE FALSE The removed Card object null ager Expected Output Card object for "Charizzard"	FALSE TRUE FALSE FALSE The removed Card object null Actual Output Card object for "Charizard"	P P P P P P P P P P P P P P P P P P P
addCard removeCard Method findCard	1 Adds a unique card to a non-full deck. 2 Falls to add a card if deck is full. 3 Falls to add a card if deck is full. 1 Removes an existing card successfully. 2 Falls to remove card at out-of-bounds index. 8 Test Description 1 Finds an existing card by name. 2 Returns null for a non-existent card. 1 Adds a new, unique card type successfully.	Add "Charmander" to deck with 5 cards. Add "Squiffe" to deck with 10 cards. Add "Telled" to deck with 10 cards. Add "Telled" to deck with 3 cards. removeCard(2) on a deck with 3 cards. Class: CollectionMan Sample Input Data findCard("Charizard") when it does not exist. Add "Subassum" for the first time.	TRUE FALSE FALSE The removed Card object null ager Expected Output Card object for "Chairzard" null TRUE	FALSE TRUE FALSE FALSE The removed Card object null Actual Output Card object for "Charizard" null TRUE	P P P P P P P P P P P P P P P P P P P
addCard removeCard Method findCard	1 Adds a unique card to a non-full deck. 2 Falls to add a card if deck is full. 3 Falls to add a card with a duplicate name. 1 Removes an existing card successfully. 2 Falls to remove card at out-of-bounds index. # Test Description 1 Finds an existing card by name. 2 Returns null for a non-existent card. 1 Adds a new, unique card type successfully. 2 Falls to add a card with a duplicate name.	Add "Charmander" to deck with 5 cards. Add "Squirlie" to deck with 10 cards. Add "Skutrie" to deck that already has a "Pikachu". removeCard(2) on a deck with 3 cards. Class: CollectionMar Sample Input Data Sample Input Data findCard("Mewtvo") when it does not exist. Add "Subasau" fog the first time. Add "Subasau" fog the first time.	TRUE FALSE FALSE The removed Card object null lager Expected Output Card object for "Charizard" null TRUE FALSE	FALSE TRUE FALSE FALSE The removed Card object null Actual Output Card object for "Charizard" null TRUE FALSE	P P P P P P P P P P P P P P P P P P P
addCard removeCard Method findCard addNewCard	Adds a unique card to a non-full deck. 2 Falls to add a card if deck is full. 3 Falls to add a card with a duplicate name. 1 Removes an existing card successfully. 2 Falls to remove card a tou-of-bounds index. 5 Test Description. 1 Finds an existing card by name. 2 Returns null for a non-existent card. 1 Adds a new. unique card type successfully. 2 Falls to add a card with a duplicate name. 1 Increases the count of an existing card.	Add "Charmander" to deck with 5 cards. Add "Squiffe" to deck with 10 cards. Add "Tellic To deck with 10 cards. Add "Tellic To deck with 10 cards. Class: CollectionMar Sample Input Data Indicard"(Charizard") when it exists. Indicard"(Charizard") when it does not exist. Add "Subassur" for the first time. Add "Subassur" again. Add "Subassur" again. Add "Subassur" again.	TRUE FALSE FALSE The removed Card object null ager Expected Output Card object for "Charizard" null TRUE FALSE true (count becomes 6)	FALSE TRUE FALSE FALSE The removed Card object null Actual Output Card object for "Charizard" null TRUE FALSE TRUE FALSE true (count becomes 6)	P P P P P P P P P P P P P P P P P P P
addCard removeCard Method findCard addNewCard increaseCount	Adds a unique card to a non-full deck. 2 Falls to add a card if deck is full. 3 Falls to add a card with a duplicate name. 1 Removes an existing card successfully. 2 Falls to remove card at out-of-bounds index.	Add "Charmander" to deck with 5 cards. Add "Squiffe" to deck with 10 cards. Add "Squiffe" to deck with 10 cards. Add "Pikachu" to deck that already has a "Pikachu". removeCard(2) on a deck with 10 cards. Class: CollectionMar Sample Input Data finGcard("Charizard") when it exists. finGcard("Charizard") when it exists. finGcard "Charizard" when it exists. Add "Bubassaur" for the first time. Add "Bubassaur" for the first time. Add "Bubassaur" for the first time. Card "A" count is 1. increaseCount("A", 5) increaseCount("S", 5) Card "C" count is 10. decreaseCount("C", 3)	TRUE FALSE FALSE The removed Card object null sager Expected Output Card object for "Charizard" null TRUE FALSE true (count becomes 6) FALSE true (count becomes 7)	FALSE TRUE FALSE FALSE The removed Card object null Actual Output Card object to "Charizard" null TRUE FALSE true (count becomes 6) FALSE	P P P P P P P P P P P P P P P P P P P
addCard removeCard Method findCard addNewCard	Adds a unique card to a non-full deck. Falist to add a card if deck is full. Falist to add a card with a duplicate name. Removes an existing card successfully. Falist to remove card at out-of-bounds index. Feet Description Finds an existing card by name. Returns null for a non-existent card. Adds a new, unique card type successfully, Falist to add a card with a duplicate name. Increases the count of an existing card. Falist to increase count of a non-existent card.	Add "Charmander" to deck with 5 cards. Add "Squirle" to deck with 10 cards. Add "Squirle" to deck that already has a "Pikachu". removeCard(2) on a deck with 3 cards. removeCard(10) on a deck with 10 cards. Class: CollectionMan Sample Input Data InnCard("Chartzard") when it exists. InnCard("Chartzard") when it does not exist. Add "Subassur" for the first time. Add "Subassur" for the first time. Add "A" count is 1. increaseCount("A", 5) increaseCount("B", 5)	TRUE	FALSE TRUE FALSE TRUE FALSE The removed Card object FALSE TRUE FALSE TRUE FALSE TRUE FALSE True (count becomes 6) FALSE True (count becomes 7) True T	P P P P P P P P P P P P P P P P P P P
memoveCard Method MindCard addNewCard ncreaseCount decreaseCount	1 Adds a unique card to a non-full deck. 2 Fals to add a card if deck is full. 3 Fals to add a card with a duplicate name. 1 Removes an existing card successfully. 2 Fals to remove card at out-of-bounds index. # Test Description 1 Finds an existing card by name. 2 Returns null for a non-existent card. 1 Adds a new, unique card type successfully. 2 Fals to add a card with a duplicate name. 1 Increases the count of an existing card. 2 Falls to increase count of a non-existent card. 1 Decreases successfully. 2 Falls to decrease count below zero. 1 Returns true if card count > 0.	Add "Charmander" to deck with 5 cards. Add "Squirle" to deck with 10 cards. Add "Squirle" to deck that already has a "Pikachu". removeCard(2) on a deck with 3 cards. **Class: CollectionMan **Sample Input Data **FincCard("Charizard") when it does not exist. Add "Dubasaur" for the first time. Add "Dubasaur" for the first time. Add "A" count is 1 increaseCount("A", 5) increaseCount("B", 5) Card "C" count is 10. decreaseCount("C", 3) Card "D" count is 2 decreaseCount("C", 5) Card "C" count is 10. decreaseCount("C", 5) Card "T" count is 10. decreaseCount("C", 5) Card "T" has count 1.	TRUE FALSE FALSE The removed Card object null ager Expected Output Card object to "Charizard" null TRUE FALSE true (count becomes 6) FALSE true (count becomes 7) false (count remains 2)	FALSE TRUE FALSE TRUE FALSE The removed Card object FALSE TRUE FALSE TRUE FALSE TRUE FALSE TRUE FALSE TRUE FALSE TRUE TRUE FALSE TRUE TRUE FALSE TRUE TRU	P P P P P P P P P P P P P P P P P P P
emoveCard Wethod indCard addNewCard ncreaseCount decreaseCount sCardAvailable	Adds a unique card to a non-full deck. 2 Falls to add a card fock is full. 3 Falls to add a card fock is full. 3 Falls to add a card fock is full. 1 Removes an existing card successfully. 2 Falls to remove card at out-of-bounds index. 1 Finds an existing card by name. 2 Returns null for a non-existent card. 1 Adds a new, unique card type successfully. 2 Falls to add a card with a duplicate name. 1 Increases the count of an existing card. 2 Falls to increase count of a non-existent card. 1 Decreases count when sufficient copies exist. 2 Falls to decrease count below zero.	Add "Charmander" to deck with 5 cards. Add "Squiffe" to deck with 10 cards. Add "This of the Control of the C	TRUE	FALSE TRUE FALSE TRUE FALSE	P P P P P P P P P P P P P P P P P P P
addCard memoveCard Method findCard addNewCard increaseCount	Adds a unique card to a non-full deck. 2 Falist to add a card if deck is full. 3 Falist to add a card with a duplicate name. 1 Removes an existing card successfully. 2 Falist to remove card at out-of-bounds index. # Test Description 1 Finds an existing card by name. 2 Returns null for a non-existent card. 1 Adds a new, unique card type successfully, 2 Falist to add a card with a duplicate name. 1 Increases the count of an existing card. 2 Falist to increase count of an existing card. 2 Falist to the count of an existing card. 2 Falist to decrease count between. 1 Returns true if card count is 0. 2 Returns false if card count is 0. 1 Returns false if card count is 0.	Add "Charmander" to deck with 5 cards. Add "Squirle" to deck with 10 cards. Add "Squirle" to deck that atreagy has a "Pikachu". removeCard(2) on a deck with 3 cards. Class: CollectionMan Sample Input Data InnCard("Charcard") when it exists. InnCard("Charcard") when it does not exist. Add "Subassur" or the first time. Add "Subassur" or the first first. Card "A" count is 1. increaseCount("A", 5) increaseCount("B", 5) Card "C" count is 10 decreaseCount("C", 3) Card "T" has count 1.	TRUE FALSE True (count becomes 6) FALSE True (count remains 2) FALSE FAL	FALSE TRUE FALSE TRUE FALSE TRUE FALSE TRUE FALSE TRUE FALSE TRUE FALSE TRUE FALSE TRUE FALSE FALS	P
emoveCard Wethod indCard addNewCard ncreaseCount decreaseCount sCardAvailable getCardTypes	Adds a unique card to a non-full deck. 2 Falls to add a card 'f deck is full. 3 Falls to add a card 'f deck is full. 3 Falls to add a card 'ell and unique set in the se	Add "Charmander" to deck with 5 cards. Add "Squiffe" to deck with 10 cards. Add "Thisachu" to deck that already has a "Pikachu". removeCard(2) on a deck with 3 cards. Class: CollectionMan Sample Input Data findCard("Chartzard") when it exists. findCard("Chartzard") when it does not exist. Add "Subasaur" again. Add "Subasaur" again. Card "A" count is 1. increaseCount("A", 5) increaseCount("B", 5) Card "D" count is 1. decreaseCount("C", 3) Card "D" count is 1. decreaseCount("D", 5) Card "C" count is 2. decreaseCount("D", 5)	TRUE FALSE True (count becomes 6) FALSE True (count remains 2) FALSE FAL	FALSE TRUE FALSE TRUE FALSE TRUE FALSE TRUE FALSE TRUE FALSE TRUE FALSE TRUE FALSE TRUE FALSE FALS	P
emoveCard Wethod indCard addNewCard ncreaseCount decreaseCount sCardAvailable getCardTypes	Adds a unique card to a non-full deck. 2 Falls to add a card 'feck is full. 3 Falls to add a card 'feck is full. 3 Falls to add a card 'elck is full. 1 Removes an existing card successfully. 2 Falls to remove card at out-of-bounds index. 5 Falls to remove card at out-of-bounds index. 6 Falls to remove card at out-of-bounds index. 7 Features multifor an out-of-bounds index. 8 Features multifor an on-existent card. 1 Adds a new. unique card type successfully. 2 Falls to add a card with a duplicate name. 1 Increases the count of an existing card. 2 Falls to increase count ven existent card. 1 Decreases count when existing card. 2 Falls to decrease count to a non-existent card. 1 Decreases count when sufficient copies exist. 2 Falls to decrease count below zero. 1 Returns true if card count is 0. 2 Returns falls in flagt count is 0. 1 Returns a list of all unique card objects. 1 Returns the correct count of a card.	Add "Charmander" to deck with 5 cards. Add "Squiffe" to deck with 10 cards. Add "Filkachu" to deck that already has a "Pikachu". removeCard(2) on a deck with 3 cards. Class: CollectionMan Sample Input Data IncCard("Men/work) when it does not exist. Add "Subassur" again. Card "A" count 1 s I increaseCount("A", 5) increaseCount("B", 5) Card "C" count 1 s CaeceaseCount("C", 3) Card "C" has count 1 s CaeceaseCount("C", 5) Card "C" has count 1. Card "F has count 1. Card "S a count 2 s CaeceaseCount("C", 5) Card "C" count 2 s CaeceaseCount("C", 5) Card "C" count 3 s CaeceaseCount("C", 5) Card "C" count 5 s CaeceaseCount("C", 5) Card "C" has count 1. Collection has 3 unique cards. Card "G" has count 1.	TRUE	FALSE TRUE FALSE TRUE FALSE TRUE FALSE TRUE FALSE TRUE FALSE TRUE FALSE TRUE FALSE TRUE FALSE FALS	P P P P P P P P P P P P P P P P P P P
emoveCard Wethod indCard addNewCard ncreaseCount decreaseCount SCardAvailable patCardTypes getCardCount	Adds a unique card to a non-full deck. 2 Falls to add a card 'feck is full. 3 Falls to add a card 'feck is full. 3 Falls to add a card 'elck is full. 1 Removes an existing card successfully. 2 Falls to remove card at out-of-bounds index. 5 Falls to remove card at out-of-bounds index. 6 Falls to remove card at out-of-bounds index. 7 Features multifor an out-of-bounds index. 8 Features multifor an on-existent card. 1 Adds a new. unique card type successfully. 2 Falls to add a card with a duplicate name. 1 Increases the count of an existing card. 2 Falls to increase count ven existent card. 1 Decreases count when existing card. 2 Falls to decrease count to a non-existent card. 1 Decreases count when sufficient copies exist. 2 Falls to decrease count below zero. 1 Returns true if card count is 0. 2 Returns falls in flagt count is 0. 1 Returns a list of all unique card objects. 1 Returns the correct count of a card.	Add "Charmander" to deck with 5 cards. Add "Squiffe" to deck with 10 cards. Add "Tikachu" to deck that already has a "Pikachu". removeCard(2) on a deck with 3 cards. Class: CollectionMar Sample Input Data Indicard("Chartzard") when it exists. Indicard("Chartzard") when it exists. Indicard("Chartzard") when it does not exist. Add "Subbasaur" for the first time. Add "Subbasaur" again. Card "A" count it 1. increaseCount("A", 5) increaseCount("B", 5) Card "Crount is 10. decreaseCount("C", 3) Card "D" count is 10. decreaseCount("C", 5) Card "C" count is 10. decreaseCount("C", 5) Card "T" has count 1. Card "F" has count 0. Collection has 3 unique cards.	TRUE	FALSE TRUE FALSE TRUE FALSE TRUE FALSE TRUE FALSE TRUE FALSE TRUE FALSE TRUE FALSE TRUE FALSE FALS	P P P P P P P P P P P P P P P P P P P
addCard memoveCard Method findCard addNewCard ncreaseCount decreaseCount sCardAvailable	Adds a unique card to a non-full deck. 2 Falls to add a card if deck is full. 3 Falls to add a card with a duplicate name. 1 Removes an existing card successfully. 2 Falls to remove card at out-of-bounds index. 7 Fest Description 1 Finds an existing card by name. 2 Returns null for a non-existent card. 1 Adds a nex; unique card type successfully. 2 Falls to add acard with a duplicate name. 1 Increases the count of an existing card. 2 Falls to increase count to fa non-existent card. 1 Decreases out to fa non-existent card. 2 Falls to decrease count below zero. 3 Returns talse if card count is 0. 1 Returns a list of all unique card objects. 3 Returns to correct of a li unique card objects. 3 Returns to correct card. 4 Returns to correct count of a card. 5 Returns to for a non-existent card. 6 Returns to for a non-existent card.	Add "Charmander" to deck with 5 cards. Add "Squiffe" to deck with 10 cards. Add "Pikachu" to deck that already has a "Pikachu". removeCard(2) on a deck with 1 cards. Class: CollectionMar Sample Input Data findCard("Chartzard") when it exists. findCard("Chartzard") when it exists. findCard "Wewbow) when it does not exist. Add "Bubassaur" for the first time. Add "Bubassaur" for the first time. Add "Bubassaur" for the first time. Card "A" count is 1. increaseCount("A", 5) increaseCount("S", 5) Card "C" count is 10. decreaseCount("C", 3) Card "D" count is 10. decreaseCount("C", 3) Card "F" has count 1. Card "F" has count 0. Collection has 3 unique cards. Card "G" has count 8. getCardCount("C") getCardCount("H") Class: BinderMana	TRUE	FALSE TRUE FALSE FALSE The removed Card object ruil Actual Output Card object for "Charizard" null TRUE FALSE THUE FALSE THUE FALSE True (count becomes 6) FALSE true (count becomes 7) false (count remains 2) TRUE FALSE TRUE FALSE TRUE FALSE	P P P P P P P P P P P P P P P P P P P
emoveCard Method indCard in	Adds a unique card to a non-full deck. 2 Falls to add a card if deck is full. 3 Falls to add a card if deck is full. 3 Falls to add a card with a duplicate name. 1 Removes an existing card successfully. 2 Falls to remove card at out-of-bounds index. # Test Description 1 Finds an existing card by name. 2 Returns null for a non-existent card. 1 Adds a new, unique card type successfully. 2 Falls to add a card with a duplicate name. 1 Increases the count of an existing card. 2 Falls to increase count of a non-existent card. 1 Decreases the count of a non-existent card. 2 Falls to increase count below zero. 1 Returns true if card count > 0. 2 Returns true if card count > 0. 2 Returns true if card count is 0. 1 Returns a list of all unique card objects. 1 Returns the correct count of a card. 2 Returns the correct count of a card.	Add "Charmander" to deck with 5 cards. Add "Squiffe" to deck with 10 cards. Add "Thischin" to deck that already has a "Pikachu". removeCard(2) on a deck with 3 cards. Class: CollectionMan Sample Input Data findCard("Charizard") when it does not exist. IndCard("Mewhon") when it does not exist. Add "Subassau" again. Card "A" count it s. IncreaseCount("A", 5) increaseCount("B", 5) Card "C" count is 10. decreaseCount("C", 3) Card "C" count is 10. decreaseCount("D", 5) Card "C" count is 2. decreaseCount("D", 5) Card "C" count is 10. decreaseCount("D", 5) Card "G" count is 10. decreaseCount("C", 3) Card "G" count is 10. decreaseCount("D", 5)	TRUE FALSE FALSE The removed Card object null agage Expected Output Card object for "Charizard" null TRUE FALSE true (count becomes 6) FALSE true (count becomes 7) Itale (count remains 2) TRUE FALSE ArrayList of size 3 Ger Expected Output	FALSE TRUE FALSE	P P P P P P P P P P P P P P P P P P P
emoveCard Method indCard inddNewCard increaseCount tecreaseCount scardAvailable getCardTypes getCardCount Method indBinder	Adds a unique card to a non-full deck. 2 Falls to add a card if deck is full. 3 Falls to add a card if deck is full. 3 Falls to add a card with a duplicate name. 1 Removes an existing card successfully. 2 Falls to remove card at out-of-bounds index. 5 Test Description. 1 Finds an existing card by name. 2 Returns null for a non-existent card. 1 Adds a new, unique card type successfully. 2 Falls to add a card with a duplicate name. 1 Increases the count of an existing card. 2 Falls to increase count with a duplicate name. 1 Increases the count of an existing card. 2 Falls to increase count when a existing card. 3 Decreases count when sufficient copies exist. 2 Falls to decrease count below zero. 1 Returns tue if card count > 0. 2 Returns false if card count is 0. 1 Returns a list of all unique card objects. 3 Returns 1 Retu	Add "Charmander" to deck with 5 cards. Add "Squiffe" to deck with 10 cards. Add "Fikachu" to deck that already has a "Pikachu". removeCard(2) on a deck with 3 cards. Class: CollectionMan Sample Input Data findCard("Chartzard") when it exists. findCard("Chartzard") when it does not exist. Add "Bubasaur" for the first time. Add "Bubasaur" again. Card "A" count is 1. increaseaCount("A", 5) increaseCount("B", 5) Card "Crount is 2. decreaseCount("C", 3) Card "D" count is 2. decreaseCount("C", 5) Card "C" has count 1. Card "A" count is 1. decreaseCount("C", 6) Card "C" card "C" count is 2. decreaseCount("C", 6) Card "C" card "C" count is 1. decreaseCount("C", 6) Card "C" card "C" can count 1. Card "C" has count 0. Collection has 3 unique cards. Card "C" card "Cutt is Count is 2. decreaseCount("C") getCardCount("H") Class: BinderMana Sample Input Data findBinder("My Trades") when it exists. findBinder("My Trades") when it exists.	TRUE	FALSE TRUE FALSE	P P P P P P P P P P P P P P P P P P P
emoveCard Method indCard inddNewCard increaseCount tecreaseCount scardAvailable getCardTypes getCardCount Method indBinder	Adds a unique card to a non-full deck. 2 Falls to add a card 'deck is full. 3 Falls to add a card 'deck is full. 1 Removes an existing card successfully. 2 Falls to remove card at out-of-bounds index. Fet Description 1 Finds an existing card by name. 2 Returns null for a non-existent card. 1 Adds a new, unique card bype successfully. 2 Falls to add a card with a duplicate name. 1 Increases the count of an existing card. 2 Falls to increase count of a non-existent card. 1 Decreases count when sufficient copies exist. 2 Falls to decrease count to fa non-existent card. 1 Decreases count when sufficient copies exist. 2 Falls to decrease count bebow zero. 1 Returns true if card count > 0. 2 Returns false if card count is 0. 1 Returns a list of all unique card objects. 1 Returns for a non-existent card. 5 Test Description 1 Finds an existing binder. 2 Returns null for non-existent binder. 1 Freats a new, unique unioner.	Add "Charmander" to deck with 5 cards. Add "Squiffe" to deck with 10 cards. Add "Fikachu" to deck that at leady has a "Pikachu". removeCard(2) on a deck with 3 cards. Class: CollectionMan Sample Input Data IndCard("Charizard") when it exists. IndCard("Charizard") when it does not exist. Add "Subbasau" again. Card "A" count is 1 increaseCount("A", 5) increaseCount("B", 5) Card "C" count is 1. GerceaseCount("D", 5) Card "C" count is 2. decreaseCount("D", 5) Card "G" count is 2. decreaseCount("G") getCardCount("H") Class: BinderMana Sample Input Data IndSinder("My Trades") when it exists. IndSinder("My Trades") when it does not exist. reaseBinder(Two Binder")	TRUE FALSE True (count becomes 6) FALSE True (count remains 2) FALSE FAL	FALSE TRUE FALSE TRUE FALSE FALSE The removed Card object rull Actual Output Card object for "Charizard" rull TRUE FALSE true (count becomes 6) FALSE true (count becomes 7) false (count remains 2) TRUE FALSE ArrayList of size 3 Actual Output Binder object	P P P P P P P P P P P P P P P P P P P
emoveCard Method indCard inddNewCard increaseCount tecreaseCount scardAvailable getCardTypes getCardCount Method indBinder	Adds a unique card to a non-full deck. 2 Falls to add a card if deck is full. 3 Falls to add a card with a duplicate name. 1 Removes an existing card successfully. 2 Falls to remove card at out-of-bounds index. 5 Test Description 1 Finds an existing card by name. 2 Returns multifor a non-existent card. 1 Adds a new. unique card type successfully. 2 Falls to add a card with a duplicate name. 1 Increases the count of an existing card. 2 Falls to increase ount of a non-existent card. 1 Decreases the count of a micrositent card. 2 Falls to decrease count below zero. 1 Returns true if card count is 0. 2 Returns falls of a did unique card objects. 3 Returns falls of a did unique card objects. 4 Returns the correct count of a card. 5 Returns for a non-existent card. 6 Returns for a non-existent card. 7 Returns of card count is 0. 8 Test Description 1 Finds an existing inder. 9 Returns a nexisting binder. 1 Finds an existing binder. 2 Returns nunque binder. 2 Falls to create a binder with a duplicate name.	Add "Charmander" to deck with 5 cards. Add "Squiffe" to deck with 10 cards. Add "Pikachu" to deck that already has a "Pikachu". removeCard(2) on a deck with 10 cards. Class: CollectionMar Sample Input Data Indianal CollectionMar Add "Bubbasaur" for the first time. Add "Bubbasaur" for the first time. Add "Bubbasaur" again. Card "A" count 1 s 1. increaseCount("A", 5) IncreaseCount("B", 5) Card "C" count 1 s 1. decreaseCount("C", 3) Card "D" count 1 s 0. decreaseCount("C", 5) Card "T" has count 1. Card "F" has count 1. Card "F" has count 1. Card "G" pagicardCount("G") getCardCount("H") Class: BinderMana Sample Input Data Indiander("Ny Trades") when it exists. Indiander("Lost Binder") when it does not exist. createBinder("New Binder") gain.	TRUE	FALSE TRUE FALSE FALSE The removed Card object rull Actual Output Card object for "Charizard" null TRUE FALSE true (count becomes 6) FALSE true (count becomes 7) false (count remains 2) TRUE FALSE TRUE FALSE TRUE FALSE TRUE FALSE TRUE FALSE FALSE FALSE ArrayList of size 3	P P P P P P P P P P P P P P P P P P P
emoveCard Method indCard addNewCard ncreaseCount decreaseCount scardAvailable petCardTypes getCardCount Method	Adds a unique card to a non-full deck. 2 Falls to add a card if eck is full. 3 Falls to add a card if eck is full. 3 Falls to add a card with a duplicate name. 1 Removes an existing card successfully. 2 Falls to remove card at out-of-bounds index. # Test Description 1 Finds an existing card by name. 2 Returns null for a non-existent card. 1 Adds a new, unique card type successfully. 2 Falls to add a card with a duplicate name. 1 Increases the count of an existing card. 2 Falls to increase count of a non-existent card. 1 Decreases the count of an existing card. 2 Falls to increase count below zero. 1 Returns true if card count > 0. 2 Returns true if card count > 0. 3 Returns a list of all unique card objects. 1 Returns the correct count of a card. 2 Returns the correct count of a card. 2 Returns the correct count of a card. 3 Returns a man the correct count of a card. 4 Returns the correct count of a card. 5 Returns null for non-existent card. 7 Returns null for non-existent binder. 1 Creates a new, unique binder. 2 Falls to create a binder with a duplicate name. 1 Deletes an existing binder.	Add "Charmander" to deck with 5 cards. Add "Squiffe" to deck with 10 cards. Add "Fikachu" to deck that already has a "Pikachu". removeCard(2) on a deck with 3 cards. Class: CollectionMan Sample Input Data findCard("Charizard") when it does not exist. findCard("Mowhon") when it does not exist. Add "Subassur" again. Card "A" count is 1. increaseCount("A", 5) increaseCount("B", 5) Card "C" count is 10. decreaseCount("C", 3) Card "C" count is 10. decreaseCount("D", 5) Card "C" count is 10. decreaseCount("C", 5) Card "G" count is 10. decreaseCount("G") Card "G" count is 10. decreaseCoun	TRUE FALSE FALSE The removed Card object null ager Expected Output Card object to "Charizard" null TRUE FALSE true (count becomes 6) FALSE true (count becomes 7) false (count remains 2) TRUE FALSE EXPECTED TRUE FALSE TRUE FALSE TRUE FALSE FALSE FALSE FALSE FALSE FALSE FALSE ArrayList of size 3	FALSE TRUE FALSE TRUE FALSE TRUE	P P P P P P P P P P P P P P P P P P P
ddCard emoveCard letethod ndCard ddNewCard ddNewCard acreaseCount lecreaseCount	Adds a unique card to a non-full deck. 2 Falls to add a card if deck is full. 3 Falls to add a card with a duplicate name. 1 Removes an existing card successfully. 2 Falls to remove card at out-of-bounds index. 5 Test Description 1 Finds an existing card by name. 2 Returns multifor a non-existent card. 1 Adds a new. unique card type successfully. 2 Falls to add a card with a duplicate name. 1 Increases the count of an existing card. 2 Falls to increase ount of a non-existent card. 1 Decreases the count of a micrositent card. 2 Falls to decrease count below zero. 1 Returns true if card count is 0. 2 Returns falls of a did unique card objects. 3 Returns falls of a did unique card objects. 4 Returns the correct count of a card. 5 Returns for a non-existent card. 6 Returns for a non-existent card. 7 Returns of card count is 0. 8 Test Description 1 Finds an existing inder. 9 Returns a nexisting binder. 1 Finds an existing binder. 2 Returns nunque binder. 2 Falls to create a binder with a duplicate name.	Add "Charmander" to deck with 5 cards. Add "Squiffe" to deck with 10 cards. Add "Pikachu" to deck that already has a "Pikachu". removeCard(2) on a deck with 10 cards. Class: CollectionMar Sample Input Data Indianal CollectionMar Add "Bubbasaur" for the first time. Add "Bubbasaur" for the first time. Add "Bubbasaur" again. Card "A" count 1 s 1. increaseCount("A", 5) IncreaseCount("B", 5) Card "C" count 1 s 1. decreaseCount("C", 3) Card "D" count 1 s 0. decreaseCount("C", 5) Card "T" has count 1. Card "F" has count 1. Card "F" has count 1. Card "G" pagicardCount("G") getCardCount("H") Class: BinderMana Sample Input Data Indiander("Ny Trades") when it exists. Indiander("Lost Binder") when it does not exist. createBinder("New Binder") gain.	TRUE	FALSE TRUE FALSE FALSE The removed Card object rull Actual Output Card object for "Charizard" null TRUE FALSE true (count becomes 6) FALSE true (count becomes 7) false (count remains 2) TRUE FALSE TRUE FALSE TRUE FALSE TRUE FALSE TRUE FALSE FALSE FALSE ArrayList of size 3	P P P P P P P P P P P P P P P P P P P

addCardToBinder		3 Fails if no copies of card are available.	Card "Zapdos" has count 0. addCardToBinder()	2 (No Copies)	2 (No Copies)	Р
		1 Removes a card from a valid index.	Remove card at index 0 from non-empty binder.	TRUE	TRUE	P
removeCardFromBinder		2 Fails to remove from an invalid index.	Remove card at index 99 from binder.	FALSE	FALSE	P
performTrade	1	Successfully completes a valid trade.	Trade card at index 0 for a new valid card.	TRUE	TRUE	P
getBinders		1 Returns a list of all binder objects.	Manager has 4 binders.	ArrayList of size 4	ArrayList of size 4	P
	•	•	Class: DeckManage		•	
Method	#	Test Description	Sample Input Data	Expected Output	Actual Output	P/F
		1 Finds an existing deck.	findDeck("Main Deck") when it exists.	Deck object	Deck object	P
findDeck		2 Returns null for non-existent deck.	findDeck("Side Deck") when it does not exist.	null	null	P
		1 Creates a new, unique deck.	createDeck("New Deck")	TRUE	TRUE	P
createDeck		2 Fails to create a deck with a duplicate name.	createDeck("New Deck") again.	FALSE	FALSE	P
		1 Deletes an existing deck.	deleteDeck("Old Deck")	TRUE	TRUE	P
deleteDeck		2 Fails to delete a non-existent deck.	deleteDeck("Non-existent Deck")	FALSE	FALSE	P
	\top	1 Adds available card to valid deck.	addCardToDeck("Pikachu", "Main Deck")	0 (Success)	0 (Success)	Р
		2 Fails if card is duplicate in deck.	Deck has "Pikachu". addCardToDeck("Pikachu",)	4 (Duplicate)	4 (Duplicate)	Р
addCardToDeck		3 Fails if deck is full.	Deck has 10 cards, addCardToDeck()	3 (Full)	3 (Full)	P
removeCardFromDeck		1 Removes a card from a valid index.	Remove card at index 0 from non-empty deck.	TRUE	TRUE	P
getDecks	+	1 Returns a list of all deck objects.	Manager has 2 decks.	ArrayList of size 2	ArrayList of size 2	P
5			Class: Validator	1	,	
Method	#	Test Description	Sample Input Data	Expected Output	Actual Output	P/F
	+	Returns true for a non-blank string.	isValidName("Test")	TRUE	TRUE	P
		2 Returns false for a blank string.	isValidName(" ")	FALSE	FALSE	P
isValidName		3 Returns false for a null string.	isValidName(null)	FALSE	FALSE	P
io validi valilo		1 Returns true for a positive double	isPositive(10.5)	TRUE	TRUE	P
	_	Returns tide for a positive double. Returns false for a negative double.	isPositive(-0.1)	FALSE	FALSE	P
		3 Returns true for a negative double.	isPositive(1)	TRUE	TRUE	- I'
isPositive		4 Returns false for a non-positive integer.	isPositive(0)	FALSE	FALSE	- 1
IOI COREVO	1	- recens also for a non-positive integer.	ioi contro(c)	IALDE	FALSE	
	1	1	Class: Inputter	I .	I .	
Method	#	Test Description	Sample Input Data	Expected Output	Actual Output	P/F
mounou	-	Returns integer on valid numeric input.	User enters "123"	Expected Output		123 P
getIntInput		2 Reprompts on invalid non-numeric input.	User enters "abo", then "45"	45		45 P
getStringInput		Returns the raw string entered by the user.	User enters " My Binder "	My Binder	My Binder	45 I
getotinginput		Returns trimmed string on valid input.	User enters " Test Name "	Test Name	Test Name	
getValidName			User enters " ", then "Valid Name"	Valid Name	Valid Name	
getvaliulvarile		2 Reprompts on blank input. 1 Returns correct enum from valid numeric selection.	User is shown list, enters "3".	Rarity.RARE	Rarity.RARE	
getValidRarity			User enters "99", then "2".	Rarity.UNCOMMON	Rarity.UNCOMMON	P P
getValidKarity getValidVariant		Reprompts on invalid numeric selection. Returns correct enum from valid numeric selection.	User enters "4".	Variant.ALT_ART	Variant.ALT_ART	P
getvaliovariant				variant.ALI_ARI 99.99	Variant.ALI_ART	99.99 P
getValidPositiveDouble		1 Returns double on valid positive input.	User enters "99.99" User enters "-10.5", then "5.5"	99.98		5.5 P
getvalidPositiveDouble		2 Reprompts on negative input.		5.5		10 P
		1 Returns int on valid positive input.	User enters "10"	10		
getValidPositiveInt	_	2 Reprompts on zero input.	User enters "0", then "1"	1		1 P
	1	T	Class: Display	Tee.	T	1
Method	#	Test Description	Sample Input Data	Expected Output	Actual Output	P/F
getHeader	+-	1 Returns a formatted header string.	getHeader("TEST")	A string with separators and the centered title "TEST"	A string with separators and the centered title "TEST"	Р
getMenu	+	1 All getMenu methods return a non-empty string.	Display.getMainMenu()	A valid menu String	A valid menu String	P P
getCardDetails		Returns a formatted string of card details. Returns a formatted, numbered list of items.	A valid Card object.	A multi-line string with all card attributes. A numbered list string with a title and 3 items.	A multi-line string with all card attributes. A numbered list string with a title and 3 items.	P P
getList	_	Returns a formatted, numbered list of items.	An ArrayList of 3 strings. Class: Handler	A numbered list string with a title and 3 items.	A numbered list string with a title and 3 items.	P
Method		Test Description		E	Actual Output	P/F
Method	#		Sample Input Data	Expected Output		P/F
		Displays collection and prompts for details view.	Collection has cards. User selects card #1.	Card #1 details are displayed.	Card #1 details are displayed.	P -
	_	2 Handles user backing out of details view.	Collection has cards. User selects "0" to go back.	No details are displayed; method returns.	No details are displayed; method returns.	P
handleViewCollection		3 Handles an empty collection gracefully.	The collection has no cards.	Prints "Collection is empty."	Prints "Collection is empty."	P
	-	Successfully adds a new, unique card.	User enters valid, unique name "Pikachu", rarity "Rare", variant "Full-art", value "25.0".	Prints "Success! Card 'Pikachu' added." and the card is added to CollectionManager.	Prints "Success! Card 'Pikachu' added." and the card is added to CollectionManager.	P
		2 Fails to add a card with a duplicate name.	User enters "Pikachu" when it already exists in the CollectionManager.	Prints "A card with this name already exists." and no new card is added.	Prints "A card with this name already exists." and no new card is added.	P
handleAddNewCard		Reprompts for a blank name and then succeeds.	User enters blank name, then "Squirtle", then valid rarity/variant/value.	Prints "Card name cannot be blank." then "Success! Card 'Squirtle' added."	Prints "Card name cannot be blank." then "Success! Card 'Squirtle' added."	P
		1 Successfully increases the count of a card.	User selects "Pikachu" (count: 2), chooses action (1) Increase, enters amount "3".	Prints "Count updated." and card count in CollectionManager becomes 5.	Prints "Count updated." and card count in CollectionManager becomes 5.	P
		2 Successfully decreases the count of a card.	User selects "Pikachu" (count: 5), chooses action (2) Decrease, enters amount "2".	Prints "Count updated." and card count in CollectionManager becomes 3.	Prints "Count updated." and card count in CollectionManager becomes 3.	- P
handlal ladata C 40 1		3 Fails to decrease count more than available.	User selects "Pikachu" (count: 3), chooses action (2) Decrease, enters amount "5".	Prints "Operation failed" and card count in CollectionManager remains 3.	Prints "Operation failed" and card count in CollectionManager remains 3.	- P
handleUpdateCardCount		Handles user backing out of selection. Displays cards in a selected binder.	User is prompted to select a card and enters "0". User selects a binder with 3 cards.	The method returns to the previous menu without changing any state. A formatted list of 3 card names is printed.	The method returns to the previous menu without changing any state. A formatted list of 3 card names is printed.	- 12
handleViewBinders	1	Displays cards in a selected binder. Handles selection of an empty binder.	User selects a binder with 3 cards. User selects a binder with 0 cards.	A formatted list of 3 card names is printed. Prints "This binder is empty."	A formatted list of 3 card names is printed. Prints "This binder is empty."	- 12
riandleviewbinders		Z manues selection of an empty binder.			r illis under is empty."	P
					Drinte PDinder greated assessed the Fond Br. 1 and Br. 1 and	
handlaCroataC'- *		Successfully creates a new binder with a unique name.	User enters a valid, unique name "Trade Binder".	Prints "Binder created successfully." and the binder is added to BinderManager.	Prints "Binder created successfully." and the binder is added to BinderManager.	P
handleCreateBinder		Successfully creates a new binder with a unique name. Fails to create a binder with a duplicate name.	User enters a valid, unique name "Trade Binder". User enters "Trade Binder" when it already exists.	Prints "Binder created successfully," and the binder is added to BinderManager. The backend prints an error message, and no new binder is added.	The backend prints an error message, and no new binder is added.	P P
		Successfully creates a new binder with a unique name. Fails to create a binder with a duplicate name. Successfully deletes an existing binder after confirmation.	User enters a valid, unique name "Trade Binder". User enters "Trade Binder" when it already exists. User selects "Trade Binder", then confirms with "yes".	Prints "Binder created successfully," and the binder is added to BinderManager. The backend prints an error message, and no new binder is added. Prints "Binder deleted." and the binder is removed from BinderManager.	The backend prints an error message, and no new binder is added. Prints "Binder deleted." and the binder is removed from BinderManager.	P P
handleCreateBinder handleDeleteBinder		Successfully creates a new binder with a unique name. Fails to create a binder with a duplicate name. Successfully deletes an existing binder after confirmation. Cancels deletion if user does not confirm.	User enters a valid, unique name "Trade Binder". User enters "Trade Binder" when it already exists. User selects "Trade Binder", then confirms with "yes". User selects "Trade Binder", then enters "no".	Prints "Binder created successfully," and the binder is added to BinderManager. The backend prints an error message, and no new binder is added. Prints "Binder deleted," and the binder is removed from BinderManager. Prints "Deletion canceled." and the binder remains.	The backend prints an error message, and no new binder is added. Prints "Binder deleted." and the binder is removed from BinderManager. Prints "Deletion canceled." and the binder remains.	P P P
handleDeleteBinder		Successfully creates a new binder with a unique name. Z fails to create binder with a unique name. Successfully deletes an existing binder after confirmation. Camois deletion if user does not confirm.	User enters a valid, unique name 'Trade Binder', User enters 'Trade Binder', when it already exists. User selects 'Trade Binder', then confirms with 'yes', User selects 'Trade Binder', then enters 'no', User selects 'Trade Binder', then enters 'no'. User selects 'Trade Binder', then selects 'Pikachur' (count > 0).	Prints Tituder created successfully" and the binder is added to BinderManager. The backend prints an error message, and no new binder is added. Prints "Binder deleted." and the binder is removed from BinderManager. Prints "Deletion canceled." and the binder menains. Prints "Carl added." and the can work from collection to binder.	The backend prints an error message, and no new binder is added. Prints "Binder deleted." and the binder is removed from BinderManager. Prints "Deleton canceled." and the binder remains. Prints "Card added." and the card moves from collection to binder.	P P P P
handleDeleteBinder handleAddCardToBinder		Successfully creates a new binder with a unique name. Fails to create a binder with a duplicate name. Usuccessfully deletes an existing binder after confirmation. Cancels deletion if user does not confirm. Successfully adds an available card to a binder. Is successfully adds an available count.	User enters a valid, unique name "Trade Binder". User enters "Trade Binder" when it already exists. User selects "Trade Binder", when confirms with "yes". User selects "Trade Binder", then enters "no". User selects "Trade Binder", then selects "Pikachu" (count > 0). User selects "Trade Binder", then selects "Newtwo" (count = 0).	Prints "Binder created successfully," and the binder is added to BinderManager. The backend prints an error message, and no new binder is added. Prints "Binder deleted," and the binder is removed from BinderManager. Prints "Deletion canceled," and the binder remains. Prints "Card added." and the card moves from collection to binder. Prints "Tailed to add card" and no state is changed.	The backend prints an error message, and no new binder is added. Prints 'Bindre deleted' and the binder is removed from BinderManager. Prints 'Deletion canceled.' and the binder remains. Prints 'Card added.' and the card moves from collection to binder. Prints 'Failed to add card.' and no state is changed.	P P P P
handleDeleteBinder	n	Successfully creates a new binder with a unique name. Fails to create a binder with a duplicate name. Successfully deletes an existing binder after confirmation. Cancels deletion if user does not confirm. Successfully adds an available card to a binder. Fails to add a card with zero available count.	User enters a valid, unique name "Trade Binder". User enters "Trade Binder" when it already exists. User selects "Trade Binder", then confirms with "yes". User selects "Trade Binder", then enters "no". User selects "Trade Binder", then selects "Pikachu" (count > 0). User selects "Trade Binder", then selects "Mewthor" (count = 0). User selects "Trade Binder", then selects "Mewthor" (count = 0).	Prints "Binder created successfully," and the binder is added to BinderManager. The backend prints an error message, and no new binder is added. Prints "Binder deleteds" and the binder is removed from BinderManager. Prints "Deletion canceled." and the binder remains. Prints "Card added." and the card moves from collection to binder. Prints "Falled to add card" and no state is changed. Prints "Card returned to collection." and the card moves from binder to collection.	The backend prints an error message, and no new binder is added. Prints "Binder deleted," and the binder is removed from BinderManager. Prints "Deletion canceled," and the binder remains. Prints "Card added," and the card moves from collection to binder. Prints "Failed to add card" and no state is changed. Prints "Card returned to collection," and the card moves from binder to collection.	P P P P P
handleDeleteBinder handleAddCardToBinder	n	Successfully creates a new binder with a unique name. Z Faits to create a binder with a duplicate name. Successfully deletes an existing binder after confirmation. C Cancels deletion if user does not confirm. Successfully deletes are existing binder after confirmation. Z Cancels deletion if user does not confirm. Z Faits to add a card with zero available count. Successfully removes a card from a binder.	User enters a valid, unique name "Trade Binder". User enters "Trade Binder when it already exists. User selects "Trade Binder", then confirms with "yes". User selects "Trade Binder", then enters "no". User selects "Trade Binder", then selects "Pikachu" (count > 0). User selects "Trade Binder", then selects "Rewhov" (count > 0). User selects "Trade Binder", then selects "Mewhov" (count = 0). User selects "Trade Binder", then selects a "Rewhov" (count = 0). User selects "Trade Binder", then selects a "Rewhov" (count = 0). User selects a "Inder selects "Rewhov" to trade away, enters valid details for a new card "Charmander".	Prints "Binder created successfully." and the binder is added to BinderManager. The backend prints an error message, and no new binder is added. Prints "Binder deleted" and the binder is removed from BinderManager. Prints "Deletion canceled." and the binder remains. Prints "Card added." and the card moves from collection to binder. Prints "Card added." and the card moves from collection to binder. Prints "Card returned to collection," and the card moves from binder to collection. Prints "Trad successfull" and "Pskach" is gone, "Chammander is now in the binder.	The backend prints an error message, and no new binder is added. Prints "Binder deleted." and the binder is removed from BinderManager. Prints "Deletion canceled." and the binder remains. Prints "Card added." and the card moves from collection to binder. Prints "Card added." and the card moves from collection to binder. Prints "Failed to add card" and no state is changed. Prints "Card returned to collection" and the card moves from binder to collection. Prints "Trade successfulf" and "Pikachu" is gone, "Charmander" is now in the binder.	P P P P P P P
handleDeleteBinder handleAddCardToBinder handleRemoveCardFromBin	n	Successfully creates a new binder with a unique name. Fails to create a binder with a duplicate name. Usuccessfully deletes an existing binder after confirmation. Canceis deletion if user does not confirm. Successfully adds an available card to a binder. Fails to add a card with zero available count. Successfully removes a card from a binder. Completes a valid trade. Completes a valid trade.	User enters a valid, unique name "Trade Binder". User enters "Trade Binder" when it already exists. User selects "Trade Binder", when confirms with "yes". User selects "Trade Binder", then enters "no". User selects "Trade Binder", then selects "Pikachu" (count > 0). User selects "Trade Binder", then selects "Mewtor" (count = 0). User selects "Trade Binder", then selects "Mewtor" (count = 0). User selects "Trade Binder", then selects a mexisting card "Pikachu" from it. User selects a binder, selects "Pikachu" to trade away, enters valid details for a new card "Charmander". Trade results in value difference > \$1.00. User enters "no" when prompted to proceed.	Prints "Binder created successfully," and the binder is added to BinderManager. The backend prints an error misesage, and no new binder is added. Prints "Binder deleted," and the binder is removed from BinderManager. Prints "Card added." and the binder remains. Prints "Card added." and the card moves from collection to binder. Prints "Card added." and no state is changed. Prints "Card returned to collection." and the card moves from binder to collection. Prints "Trade successfull" and "Pikachu" is gone, "Charmander" is now in the binder. Prints "Trade successfull" and "Pikachu" is gone, "Charmander" is now in the binder.	The backend prints an error message, and no new binder is added. Prints 'Binder deleted,' and the binder is removed from BinderManager. Prints 'Deletion canceled,' and the binder remains. Prints 'Card added,' and the card moves from collection to binder. Prints 'Failed to add card.' and no state is changed. Prints 'Card enturned to collection.' and the card moves from binder to collection. Prints 'Trade auccessful' and 'Pikachu' is gone, 'Chammander' is now in the binder. Prints 'Trade auccessful' and on state is changed.	P P P P P P P P P P P P P P P P P P P
handleDeleteBinder handleAddCardToBinder handleRemoveCardFromBin handleTradeFromBinder	n	1 Successfully creates a new binder with a unique name. 2 Fails to create binder with a unique name. 1 Successfully deletes an existing binder after confirmation. 2 Cannois deletion if user obes not confirm. 1 Successfully adds an available cond to a binder. 2 Fails to add a card with zero available count. 1 Successfully adds an available count. 1 Completes a valid trade. 2 Cannois an unfair trade upon user prompt. 3 Fails if incomplic and details are invalid.	User enters a valid, unique name 'Trade Binder', User enters 'Trade Binder', when it already exists. User elects 'Trade Binder', then confirms with 'yes'. User selects 'Trade Binder', then enters 'no'. User selects 'Trade Binder', then selects 'Pikachu' (count > 0). User selects 'Trade Binder', then selects 'Pikachu' (count > 0). User selects 'Trade Binder', then selects 'Pikachu' (count > 0). User selects 'Trade Binder', then selects an existing card 'Pikachu' from it. User selects a binder, selects 'Pikachu' to trade away, enters valid details for a new card 'Charmander'. Trade results in value difference >= \$1.00. User enters 'no' when prompted to proceed. User enters a blank name for the incoming card.	Prints Trade canceled "accessfully" and the binder is added to BinderManager. The backend prints an error message, and no new binder is added. Prints "Binder deleted." and the binder is removed from BinderManager. Prints "Carl daded." and the chinder remains. Prints "Carl daded." and the card moves from collection to binder. Prints "Tailed to add card" and no state is changed. Prints "Tailed to add card" and no state is changed. Prints "Triade successfull" and "Pikachu" is gone, "Chammander' is now in the binder. Prints "Trade canceled." and no state is changed. Prints "Trade canceled." and no state is changed. Prints "Trade canceled." and no state is changed.	The backend prints an error message, and no new binder is added. Prints "Binder deleted," and the binder is removed from BinderManager. Prints "Deletion canceled," and the binder remains. Prints "Card added," and the card moves from collection to binder. Prints "Failed to add card" and no state is changed. Prints "Card returned to collection," and the card moves from binder to collection. Prints "Trade successful!" and "Pikachu" is gone, "Charmander' is now in the binder. Prints "Irade canceled." and no state is changed. Prints "Irade canceled." and no state is changed.	P P P P P P P P P P P P P P P P P P P
handleDeleteBinder handleAddCardToBinder handleRemoveCardFromBin	n	1 Successfully creates a new binder with a unique name. 2 Fails to create a binder with a duplicate name. 1 Successfully deletes an existing binder after confirmation. 2 Canceis deletion if user does not confirm. 1 Successfully deletes an existing binder after confirmation. 2 Fails to add a card with zero available count. 1 Successfully removes a card from a binder. 2 Fails to add a card with zero available count. 3 Completes a valid trade. 2 Canceis an unflair trade upon user prompt. 3 Fails if incoming card details are invalid. 1 Successfully creates a new deck.	User enters a valid, unique name "Trade Binder". User enters "Trade Binder" when it already exists. User selects "Trade Binder" when confirms with "yes". User selects "Trade Binder", then onters "no". User selects "Trade Binder", then selects "Pikachy" (count > 0). User selects "Trade Binder", then selects "Pikachy" (count > 0). User selects "Trade Binder", then selects "Newtwo" (count = 0). User selects "Trade Binder", then selects "Newtwo" (count = 0). User selects bunder, then selects "Newtwo" (count = 0). User selects bunder, selects "Readur" to Irade away, enters valid details for a new card "Charmander". Trade results in value difference >= \$1.00. User enters "no" when prompted to proceed. User enters a blank name for the incoming card.	Prints "Binder created successfully." and the binder is added to BinderManager. The backend prints an error message, and no new binder is added. Prints "Binder detects" and the binder is removed from BinderManager. Prints "Deletion canceled." and the binder remains. Prints "Carl added." and the card moves from collection to binder. Prints "Carl added." and no state is changed. Prints "Carl entermed to collection." and the card moves from binder to collection. Prints "Trade canceled." and no state is changed. Prints "Trade canceled." and no state is changed. Prints "Trincoming card name cannot be blank." and the trade is aborted. Prints "Trincoming card name cannot be blank." and the trade is aborted.	The backend prints an error message, and no new binder is added. Prints "Binder deleted" and the binder is removed from BinderManager. Prints "Deletion canceled." and the binder remains. Prints "Card added." and the card moves from collection to binder. Prints "Failed to add card" and no state is changed. Prints "Gard returned to collection." and the card moves from binder to collection. Prints "Trade canceled." and no state is changed. Prints "Trade canceled." and no state is changed. Prints "Incommig card name cannot be blank". and the trade is aborted. Prints "Deck receded successfully" and "Beard to decked to bedManager.	P P P P P P P P P P P P P P P P P P P
handleDeleteBinder handleAddCardToBinder handleRemoveCardFromBin handleTradeFromBinder handleCreateDeck	n	1 Successfuly creates a new binder with a unique name. 2 Fails to create a binder with a unique name. 1 Successfully deletes an existing binder after confirmation. 2 Locates deletion it seed to see to confirm. 1 Successfully adds an available cond to a binder. 2 Fails to add a card with zero available count. 1 Successfully adds an available count. 1 Completes a valid trade. 2 Canalis an unifor trade upon user prompt. 3 Fails if incoming card details are invalid. 1 Successfully creates a new deck. 1 Fails to add a cupicate card to a deck.	User enters a valid, unique name 'Trade Binder', User enters 'Trade Binder', when it already exists. User enters 'Trade Binder', then confirms with 'yes'. User selects 'Trade Binder', then enters 'no'. User selects 'Trade Binder', then enters 'no'. User selects 'Trade Binder', then selects 'Pikachu' (count > 0). User selects 'Trade Binder', then selects 'Pikachu' (count > 0). User selects 'Trade Binder', then selects an existing card 'Pikachu' from it. User selects a binder, selects an existing card 'Pikachu' from it. User selects a binder, selects 'Pikachu' to trade away, enters valid details for a new card 'Charmander'. Trade results in value difference >= \$1.00. User enters 'no' when prompted to proceed. User enters a blank name for the incoming card. User enters a valid, unique name 'Main Deck'. User selects 'Main Deck' (already contains 'Pikachu'), then selects 'Pikachu' to add.	Prints Tinder created successfully" and the binder is added to Bindershanager. The backend prints an error message, and no new binder is added. Prints Tinder deleted." and the binder is removed from Bindershanager. Prints Tinder deleted." and the binder menans. Prints Tinder deleted." and the binder menans. Prints Tinder deleted." and the adverse from collection to binder. Prints Tinder adverse to added." and the card moves from binder to collection. Prints Tinde successfull and "Pikachtu" is gone. "Chammander" is now in the binder. Prints Tinde auccessfull and "Pikachtu" is gone. "Chammander" is now in the binder. Prints Tinde auccessfull and "Bikachtu" is gone. "Chammander" is now in the binder. Prints Tinde auccessfull and passe is changed. Prints Tinde auccessfull and catellies changed. Prints Tinde auccessfull and catellies can be changed. Prints Tinder auccessfull and catellies can be changed.	The backend prints an error message, and no new binder is added. Prints 'Bulded deleted,' and the binder is removed from BinderManager. Prints 'Deletion canceled,' and the binder remains. Prints 'Card added,' and the card moves from collection to binder. Prints 'Card added,' and the card moves from collection to binder. Prints 'Tarde added,' and the state is changed. Prints 'Card returned to collection,' and the card moves from binder to collection. Prints 'Trade acucessfull' and 'Pikachu' is gone, 'Chammander' is now in the binder. Prints 'Trade canceled,' and no state is changed. Prints 'Trade canceled,' and no state is changed. Prints 'Trade canceled,' and no state is changed. Prints 'Tock created successfully,' and the deck is added to DeckManager. Prints 'Talded cand cand. (or duplicate card)' 'State is unchanged.	P P P P P P P P P P P P P P P P P P P
handleDeleteBinder handleAddCardToBinder handleRemoveCardFromBin handleTradeFromBinder handleCreateDeck handleAddCardToDeck	n	1 Successfully creates a new binder with a unique name. 2 Fails to create a binder with a duplicate name. 1 Successfully deletes an existing binder after confirmation. 2 Cancels deletion if user does not confirm. 1 Successfully deletes are existing binder after confirmation. 2 Fails to add a card with zero available count. 1 Successfully removes a card from a binder; 1 Completes a valid trade. 2 Cancels an unfair trade upon user prompt. 3 Fails if incoming card details are invalid. 1 Successfully creates a new deck. 2 Fails to add a duplicate card to a deck. 2 Fails Successfully deads a unique card to a deck. 2 Fails Successfully deads a unique card to a deck.	User enters a valid, unique name "Trade Binder". User enters "Trade Binder" when it already exists. User selects "Trade Binder", then confirms with "yes". User selects "Trade Binder", then confirms vith "yes". User selects "Trade Binder", then selects "Review" (count > 0). User selects "Trade Binder", then selects "Review" (count > 0). User selects "Trade Binder", then selects "Review" (count > 0). User selects "Trade Binder", then selects "Review" (count > 0). User selects "Trade Binder", then selects "Review" (count > 0). User selects "Bander, selects "Review" enters valid details for a new card "Charmander". Trade results in value difference >= \$1.00. User enters "no" when prompted to proceed. User enters a valid, unique name "Main Deck". User selects "Main Deck", (already contains "Pliachu"), then selects "Pikachu" to add. User selects "Main Deck", (already contains "Pliachu"), then selects "Pikachu" to add.	Prints *TBinder created successfully.** and the binder is added to BinderManager. The backend prints an error message, and no new binder is added. Prints *Tbinder deleted.** and the binder is removed from BinderManager. Prints *Toleletion canceled.** and the binder remains. Prints *Toleletion canceled.** and the binder remains. Prints *Tailed to add card* and no state is changed. Prints *Tailed to add card* and no state is changed. Prints *Trade successfull* and *Pikachur is gone, *Chammander is now in the binder. Prints *Trade canceled.** and no state is changed. Prints *Trade canceled.** and sate is changed. Prints *Tailed to add card or duplicate cardy.* State is unchanged. Prints *Tailed code.** and the card wows form collection to deck.	The backend prints an error message, and no new binder is added. Prints "Burder deleted" and the binder is removed from BinderManager. Prints "Deletion canceled." and the binder remains. Prints "Card added." and the arrd moves from collection to binder. Prints "Failed to add card." and no state is changed. Prints "Gard returned to collection" and the card moves from binder to collection. Prints "Trade canceled." and no state is changed. Prints "Pailed to add card or duplicate cardy." State is unchanged. Prints "Bailed to add card or duplicate cardy." State is unchanged. Prints "Pailed to add card or duplicate cardy." State is unchanged.	P P P P P P P P P P P P P P P P P P P
handleDeleteBinder handleAddCardToBinder handleRemoveCardFromBin handleTradeFromBinder handleCreateDeck handleAddCardToDeck handleAddCardToDeck handleRemoveCardFromDeck	n	Successfully creates a new binder with a unique name. Falls to create a binder with a unique name. Successfully deletes an existing binder after confirmation. Cancels deletion if user does not confirm. Successfully adds an available card to a binder. Successfully adds an available card to a binder. Successfully adds an available card to a binder. I Falls to add a card with zero available count. Successfully read on a binder. Cancels an unifer vade upon user prompt. Services a valid trade. Cancels an unifer vade upon user prompt. Servicessfully creates a new deck. Falls to add a duplicate card to a deck. Successfully adds a unique card to a deck. Successfully adds a unique card to a deck.	User enters a valid, unique name "Trade Binder". User enters "Trade Binder" when it already exists. User selects "Trade Binder" when it already exists. User selects "Trade Binder", then confirms with "yes". User selects "Trade Binder", then enters "no". User selects "Trade Binder", then selects "Pikachu" (count > 0). User selects "Trade Binder", then selects "Newtwo" (count = 0). User selects "Trade Binder", then selects "Newtwo" (count = 0). User selects Trade Binder", then selects an existing card "Pikachu" from it. User selects a binder, selects "Readur" to rade away, enters valid details for a new card "Charmander". Trade results in value difference >= \$1.00. User enters "no" when prompted to proceed. User enters a blank name for the incoming card. User enters a valid, unique name "Main Deck". User selects "Main Deck" (already contains "Pikachu"), then selects "Pikachu" to add. User selects "Main Deck", then selects "Squirtler" (which is not in the deck). User selects "Main Deck", then selects "Squirtler" (which is not in the deck).	Prints "Binder created successfully," and the binder is added to BinderManager. The backend prints an error message, and no new binder is added. Prints "Binder detects" and the binder is removed from BinderManager. Prints "Deletion canceled," and the binder remains. Prints "Card added," and the card moves from collection to binder. Prints "Card added," and the card moves from collection to binder. Prints "Card returned to collection," and the card moves from binder to collection. Prints "Trade canceled," and no state is changed. Prints "Trade added," and the state is added to DeckManager. Prints "Trailed to add card,", or duplicate card," State is unchanged. Prints "Trailed or add card,", or duplicate card," State is unchanged. Prints "Card added," and the card moves from collection to deck. Prints "Card added and the card moves from collection to deck. Prints "Card emoved and returned to collection."	The backend prints an error message, and no new binder is added. Prints "Binder deleted," and the binder is removed from BinderManager. Prints "Deleton canceled," and the binder remains. Prints "Card added," and the card moves from collection to binder. Prints "Failed to add card" and no state is changed. Prints "Gard returned to collection." and the card moves from binder to collection. Prints "Trade successful" and "Pikachu" is gone, "Charmander" is now in the binder. Prints "Trade canceled." and no state is changed. Prints "Trade canceled." and no state is changed. Prints "Trade added." and the card moves from binder to collection. Prints "Trade added." and the state is added to DeckManager. Prints "Failed to add card" (or duplicate card). State is unchanged. Prints "Card added." and the card moves from collection to deck. Prints "Card added." and the card moves from collection to deck.	P P P P P P P P P P P P P P P P P P P
handleDeleteBinder handleAddCardToBinder handleRemoveCardFromBin handleTradeFromBinder handleCreateDeck handleAddCardToDeck	n	1 Successfully creates a new binder with a unique name. 2 Fails to create binder with a unique name. 1 Successfully deletes an existing binder after confirmation. 2 Cannotis deletion if user does not confirm. 1 Successfully adds an available count of a binder. 2 Fails to add a card with zero available count. 1 Successfully removes a card from a binder. 1 Completes a valid trade. 2 Cannotis an unfair trade upon user prompt. 3 Fails if nomoning card details are invalid. 1 Successfully creates a new deck. 2 Successfully radds a unique card to a deck. 2 Successfully adds a unique card to a deck. 1 Successfully removes a card from a deck. 1 Successfully removes a card from a deck.	User enters a valid, unique name 'Trade Binder', User enters 'Trade Binder', when it already exists. User ealects 'Trade Binder', then confirms with 'yes'. User selects 'Trade Binder', then enters 'no'. User selects 'Trade Binder', then selects 'Pikachu' (count > 0). User selects 'Trade Binder', then selects 'Pikachu' (count > 0). User selects 'Trade Binder', then selects 'no', then valid of the selects' and 'Pikachu' from it. User selects 'Trade Binder', then selects an existing card 'Pikachu' from it. User selects a binder, selects 'Pikachu' to trade away, enters valid details for a new card 'Charmander'. Trade results in value difference >= \$1.00. User enters 'no' when prompted to proceed. User enters a valid, unique name 'Main Deck'. User selects 'Main Deck', then selects 'Squirtler' (which is not in the deck). User selects 'Main Deck', then a card from its list. User selects 'Main Deck', then a card from its list. User selects 'Main Deck', then a card from its list.	Prints Trade canceled "and the black" and the brinder is added to BinderManager. The backend prints a meror message, and no new binder is added. Prints "Binder deleted." and the binder is removed from BinderManager. Prints "Deletion canceled." and the binder remains. Prints "Card added." and the card moves from collection to binder. Prints "Tailed to add card." and no state is changed. Prints "Trade canceled." and the card moves from binder to collection. Prints "Trade canceled." and no state is changed. Prints "Trade canceled." and no state is changed. Prints "Trade canceled." and no state is changed. Prints "Deck created successfully" and prilace to blank." and the trade is aborted. Prints "Deck created successfully" and the deck is added to DeckManager. Prints "Eadled and card (" or duplicate card)" State is unchanged. Prints "Card added." and the card moves from collection to deck. Prints "Card removed and returned to collection." Prints "Card removed and returned to collection."	The backend prints an error message, and no new binder is added. Prints 'Brinder detects' and the binder is removed from BinderManager. Prints 'Card addeds' and the binder remains. Prints 'Card addeds' and the card moves from collection to binder. Prints 'Card addeds' and the card moves from collection to binder. Prints 'Card returned to collection' and the card moves from binder to collection. Prints 'Trade canceled' and no state is changed. Prints 'Trade canceled' and no state is changed. Prints 'Trade card from card no state is changed. Prints 'Deck created successfully' and the deck is added to DecManager. Prints 'Teale dadd' and the card moves from collection to deck. Prints 'Card added' and the card moves from collection to deck. Prints 'Card removed and returned to collection'. Prints 'Card removed and returned to collection'.	P P P P P P P P P P P P P P P P P P P
handleDeleteBinder handleAddCardToBinder handleRemoveCardFromBin handleTradeFromBinder handleCreateDeck handleAddCardToDeck handleAddCardToDeck handleRemoveCardFromDeck	n	1 Successfully creates a new binder with a unique name. 2 Fails to create a binder with a duplicate name. 1 Successfully deletes an existing binder after confirmation. 2 Cancois deletion if user does not confirm. 1 Successfully deletes an existing binder after confirmation. 2 Fails to add a card with zero available count. 2 Fails to add a card with zero available count. 1 Successfully removes a card from a binder: 1 Completes a vail trade. 2 Canceis an unfair trade upon user prompt. 3 Failst if incoming card delatis are invalid. 1 Successfully creates a new deck. 1 Fails to add a duplicate card to a deck. 2 Successfully greates an example of the card to a deck. 1 Successfully removes a card from a deck. 1 Successfully removes a card from a deck. 1 Successfully deletes a deck after confirmation. 1 Returns the correct Card object on valid selection.	User enters a valid, unique name "Trade Binder", User enters "Trade Binder" when it already exists. User selects "Trade Binder", when confirms with "yes". User selects "Trade Binder", then enters "no". User selects "Trade Binder", then selects "Rekarbu' (count > 0). User selects "Trade Binder", then selects "Rekarbu' (count > 0). User selects "Trade Binder", then selects "Rekarbu' (count > 0). User selects "Trade Binder", then selects are avisiting card "Pikachu" from it. User selects a binder, selects "Rekarbu' to trade away, enters valid details for a new card "Charmander". Trade results in value difference >= \$1.00. User enters "no" when prompted to proceed. User enters a blank name for the incoming card. User enters a valid, unique name "Main Deck". User selects "Main Deck" (already contains "Pikachu"), then selects "Pikachu" to add. User selects "Main Deck", then a card from its list. User selects "Main Deck", then a card from its list. User selects "Main Deck", then a card from its list. User selects "Main Deck", then a main collection list.	Prints "Binder created successfully." and the binder is added to BinderManager. The backend prints an error message, and no new binder is added. Prints "Binder detects" and the binder is removed from BinderManager. Prints "Deletion canceled." and the binder remains. Prints "Card added." and the card moves from collection to binder. Prints "Failed to add card" and no state is changed. Prints "Tarde canceled." and no state is changed. Prints "Trade canceled." and no state is changed. Prints "Prade canceled." and search and the trade is aborted. Prints "Card dead" and the card moves from collection to deck. Prints "Card removed and returned to collection." Prints "Card develock deleted." The Card object corresponding to the second item in the sorted list is returned.	The backend prints an error message, and no new binder is added. Prints "Bunder deleted" and the binder is removed from BinderManager. Prints "Deletion canceled." and the binder remains. Prints "Card added." and the card moves from collection to binder. Prints "Failed to add card" and no state is changed. Prints "Failed to add card" and no state is changed. Prints "Trade canceled." and no state is changed. Prints "Trade canceled." and no state is changed. Prints "Trade canceled." and no state is changed. Prints "Failed to add card and on state is changed. Prints "Pailed to add card or duplicate card)". State is unchanged. Prints "Card reded successfully" and the deck is added to DecMinanger. Prints "Deck created successfully" and the deck is added to DecMinanger. Prints "Card added" and the card moves from collection to deck. Prints "Card carded" and the card moves from collection to deck. Prints "Card added" and the card moves from collection to the Card Deletion to the Card Deletion of the C	P P P P P P P P P P P P P P P P P P P
handleDeleteBinder handleAddCardToBinder handleRemoveCardFromBin handleTradeFromBinder handleTradeFromBinder handleCateBebeck handleAddCardToBeck handleETendErfromDelethandleEtendErfromDelethandleDeleteDeck	n	1 Successfully creates a new binder with a unique name. 2 Fails to create a binder with a unique name. 1 Successfully deletes an existing binder after confirmation. 2 Cancels obletion it user does not confirm. 1 Successfully adds an available cond to a binder. 2 Fails to add a card with zero available count. 1 Successfully removes a card from a binder. 1 Completes a valid trade. 2 Cancels an unfair trade upon user prompt. 3 Fails if incoming card details are invalid. 1 Successfully creates a new deck. 1 Successfully creates and existing the card of a deck. 2 Successfully deletes a deck after confirmation. 1 Successfully removes a card from a deck. 1 Successfully deletes a deck after confirmation. 1 Returns the correct Card object on valid selection. 2 Returns multiple user op to plack.	User enters a valid, unique name 'Trade Binder', User enters 'Trade Binder', when it already exists. User elects 'Trade Binder', then confirms with 'yes'. User selects 'Trade Binder', then enters 'no'. User selects 'Trade Binder', then enters 'no'. User selects 'Trade Binder', then selects 'Pikachu' (count > 0). User selects 'Trade Binder', then selects 'Pikachu' (count > 0). User selects 'Trade Binder', then selects a rexisting card 'Pikachu' from it. User selects a binder, selects 'Pikachu' to trade away, enters valid details for a new card 'Charmander'. Trade results in value difference >= \$1.00. User enters 'no' when prompted to proceed. User enters a valid, unique name 'Main Deck'. User selects 'Main Deck', 'then selects 'Sekachu', then selects 'Pikachu' to add. User selects 'Main Deck', 'then a card from its list. User selects 'Main Deck', 'then a confirms with 'yes'.' User selects 'Main Deck', 'then a confirms with 'yes'.' User selects 'Main Deck', 'then a card from its list. User selects card its 'Rom the main collection list. User selects and its 'Nor the main collection list. User selects card its 'Rom the main collection list.	Prints Tirder created successfully" and the binder is added to BinderManager. The backend prints an error message, and no new binder is added. Prints "Binder deleted." and the binder is removed from BinderManager. Prints "Card added." and the center is added. Prints "Card added." and the card moves from collection to binder. Prints "Tailed to add card" and no state is changed. Prints "Tailed to add card" and no state is changed. Prints "Trade canceled." and no state is changed. Prints "Card added." and the card moves from collection to deck. Prints "Card added." and the card moves from collection to deck. Prints "Card added." and the card moves from collection to deck. Prints "Card added." and the card moves from collection to deck. Prints "Card added." and the card moves from collection to deck. Prints "Card added." and the card moves from collection." Prints "Card added." and the card moves from collection." Prints "Card added." and the card moves from tollection." Prints "Card added." and the card moves from tollection."	The backend prints an error message, and no new binder is added. Prints 'Dieloted deleted,' and the binder is removed from BinderManager. Prints 'Deleton canceled,' and the binder remains. Prints 'Card added,' and the card moves from collection to binder. Prints 'Failed to add card' and no state is changed. Prints 'Tarde successfull' and 'Pikachu' is gone, 'Chammander' is now in the binder. Prints 'Trade acucessfull' and 'Pikachu' is gone, 'Chammander' is now in the binder. Prints 'Trade canceled' and no state is changed. Prints 'Trade canceled' and no state is changed. Prints 'Trade acucestfull' and 'Pikachu' is gone, 'Chammander' is now in the binder. Prints 'Trade successfully' and the deck is added to DecManager. Prints 'Trade readest successfully' and the deck is added to DecManager. Prints 'Trade readest successfully' and the deck is added to DecManager. Prints 'Trade readest successfully' and the deck is added to DecManager. Prints 'Trade readest successfully' and the deck is added to DecManager. Prints 'Trade successfully' and the deck is added to DecManager. Prints 'Trade successfully' and the deck is added to DecManager. Prints 'Trade successfully' and the deck is added to DecManager. Prints 'Trade successfully' and the deck is added to DecManager. Prints 'Trade successfully' and the deck is added to DecManager. Prints 'Trade successfully' and the deck is added to DecManager. Prints 'Trade successfully' and the deck is added to DecManager. Prints 'Trade successfully' and the deck is added to DecManager. Prints 'Trade successfully' and the deck is added to DecManager. Prints 'Trade successfully' and the deck is added to DecManager. Prints 'Trade successfully' and the deck is added to DecManager. Prints 'Trade successfully' and 'Pikachu' is gone, 'Chammanage' is now in the intervent of the successfully' and 'Pikachu' is gone, 'Chammanage' is now in the intervent of the successfully is now in the intervent of the successfully is now in the intervent of the successfully is now	P P P P P P P P P P P P P P P P P P P
handleDeleteBinder handleAddCardToBinder handleRemoveCardFromBin handleTradeFromBinder handleCreateDeck handleAddCardToDeck handleAddCardToDeck handleRemoveCardFromDeck	n	1 Successfully creates a new binder with a unique name. 2 Fails to create a binder with a duplicate name. 1 Successfully deletes an existing binder after confirmation. 2 Cannotis deletion if user does not confirm. 1 Successfully deletes an existing binder after confirmation. 2 Fails to add a card with zero available count. 1 Successfully removes a card from a binder. 1 Completes a valid trade. 2 Cannotis an unfair trade upon user prompt. 3 Fails if incoming card details are invalid. 1 Successfully creates a new deck. 1 Successfully removes a card from a deck. 2 Successfully removes a card from a deck. 1 Successfully removes a card from a deck. 1 Successfully removes a card from a deck. 2 Returns the correct Card object on valid selection. 2 Returns null when user chooses to go back. 3 Handles an empty collection gracefully.	User enters a valid, unique name "Trade Binder". User enters "Trade Binder" when it already exists. User selects "Trade Binder", then confirms with "yes". User selects "Trade Binder", then enters "no". User selects "Trade Binder", then selects "Pikachu" (count > 0). User selects "Trade Binder", then selects "Rewithor" (count = 0). User selects "Trade Binder", then selects "Mewithor" (count = 0). User selects "Trade Binder", then selects "Mewithor" (count = 0). User selects "Trade Binder", then selects a "well of the selects a binder, selects "Binder" to trade selects "Trade Binder", then selects "Abenture to results in value difference >= \$1.00. User enters and details for a new card "Charmander". Trade results in value difference >= \$1.00. User enters "no" when prompted to proceed. User enters a valid, unique name Main Deck". User enters a valid, unique name Main Deck". User selects "Main Deck", (already contains "Pikachu"), then selects "Pikachu" to add. User selects "Main Deck", then a card from its list. User selects "Main Deck", then a card from its list. User selects and #2 from the main collection list. User selects option "D". User selects option "D". Method is caled when no cards exist.	Prints Timuter created successfully." and the binder is added to BinderManager. The backend prints an error message, and no new binder is added. Prints "Binder detects" and the binder is removed from BinderManager. Prints "Deletion canceled." and the binder remains. Prints "Card added." and the card waves from collection to binder. Prints "Tailed to add card" and no state is changed. Prints "Tailed to add card" and no state is changed. Prints "Trade canceled." and no state is changed. Prints "Trade dear dear of the state is changed. Prints "Trade dear dear of the state is changed. Prints "Trade dear dear of the state is changed. Prints "Trade deard." and the card moves from collection to deck. Prints "Card removed and returned to collection." Prints "Card celected." The Card object corresponding to the second item in the sorted list is returned. null is returned.	The backend prints an error message, and no new binder is added. Prints "Burder deleted" and the binder is removed from BinderManager. Prints "Cale deleted" and the binder remains. Prints "Cale and canded" and the binder remains. Prints "Card added" and the card moves from collection to binder. Prints "Faide to add card." and no state is changed. Prints "Trade cancelad" and "Reburbur is gone, "Charmander" is now in the binder. Prints "Trade cancelad" and no state is changed. Prints "Trade cancelad" and no state is changed. Prints "Prints "Card added" and the card moves from binder to collection. Prints "Prints "Date Card name cannot be blank" and the trade is aborted. Prints "Pailed to add card." or duplicate cardy." State is unchanged. Prints "Card added" and the card moves from collection to deck. Prints "Card removed and returned to collection." The Card object corresponding to the second item in the sorted list is returned. null is returned. The Card object corresponding to the second item in the sorted list is returned. null is returned.	P P P P P P P P P P P P P P P P P P P
handleDeleteBinder handleAddCardToBinder handleRemoveCardFromBin handleTradeFromBinder handleCradeFoek handleAddCardToBick handleAddCardToBick handleAddCardToBick selectCardFromDes selectCardFromCollection	n	1 Successfully creates a new binder with a unique name. 2 Fails to create a binder with a unique name. 1 Successfully deletes an existing binder after confirmation. 2 Cancels deletion if user does not confirm. 1 Successfully adds an available count of a binder. 2 Fails to add a card with zero available count. 3 Successfully adds an available count. 1 Successfully adds an available count. 1 Completes a valid trade. 2 Cancels an unifer trade upon user prompt. 3 Fails if incoming card details are invalid. 1 Successfully reates a new deck. 1 Successfully reates an ew deck. 2 Successfully adds a unique card to a deck. 2 Successfully adds a unique card to a deck. 1 Successfully deletes a deck after confirmation. 1 Returns the correct Card do bject on valid selection. 2 Returns nutl when user chooses to go back. 3 Handles an empty collection gracefully. 1 Returns the orrect Card do sprovided list.	User enters a valid, unique name "Trade Binder", User enters "Trade Binder", when it already exists. User selects "Trade Binder", then confirms with 'yes"; User selects "Trade Binder", then enters "no". User selects "Trade Binder", then enters "no". User selects "Trade Binder", then selects "Pikachu" (count > 0). User selects "Trade Binder", then selects "Pikachu" (count > 0). User selects "Trade Binder", then selects "Revervo" (count = 0). User selects a binder, selects a meximic grader "Pikachu" from it. User selects a binder, selects "Pikachu" to trade away, enters valid details for a new card "Charmander". Trade results in value difference >= \$1.00. User enters "no" when prompted to proceed. User enters a blank name for the incoming card. User enters a valid, unique name "Main Deck". User selects "Main Deck", then selects "Spikachu", then selects "Pikachu" to add. User selects "Main Deck", then selects "Spikachu", then selects "Pikachu" to add. User selects "Main Deck", then selects "Spikachu", then selects "Pikachu" to add. User selects "Main Deck", then selects "Spikachu", then selects "Pikachu" to add. User selects "Main Deck", then selects "Spikachu", then selects "Pikachu" to add. User selects "Main Deck", then selects "Spikachu", then selects "Pikachu" to add. User selects "Main Deck", then selects "Spikachu", then selects "Pikachu" to add. User selects "Main Deck", then selects "Spikachu", then selects "Pikachu" to add. User selects "Main Deck" then card from its list. User selects selects selects "Rom the main collection list. User selects selects selects selects selects selects "Rom the main collection list. User selects selects selects selects selects selects selects selects "Rom the main collection list.	Prints Timuder created successfully" and the binder is added to Bindershanager. The backend prints an error message, and no new binder is added. Prints Tinder deleted." and the binder is removed from Bindershanager. Prints Tinder deleted." and the binder is removed from Bindershanager. Prints Tinder deleted." and the binder remains. Prints Tinder and and the tinder remains. Prints Tinder and and the tinder tinder to collection to binder. Prints Tinde successfull and "Pikachtu' is gone, "Chammander' is now in the binder. Prints Trade successfull and "Pikachtu' is gone, "Chammander' is now in the binder. Prints Tinde ancessfull and "Pikachtu' is gone, "Chammander' is now in the binder. Prints Tinde ancessfull and "Pikachtu' is gone, "Chammander' is now in the binder. Prints Tinde canceled." and no state is changed. Prints Tinde and cand cand (or dyticate card)", State is unchanged. Prints Tinder and cand cand (or dyticate card)", State is unchanged. Prints Tinder burder and the card moves from collection to deck. Prints Tinder burder and the card moves from collection to deck. Prints Tinder burder and the card moves from collection. Prints Tinder burder and the card moves from collection. Prints Tinder burder and the card moves from collection. Prints Tinder burder and the card moves from collection. Prints Tinder burder and the card moves from collection. Prints Tinder burder and the card moves from collection. Prints Tinder burder and the card moves from collection. Prints Tinder burder and the card moves from collection. Prints Tinder burder and the card moves from collection. Prints Tinder burder and the card moves from collection. Prints Tinder burder and the card moves from collection. Prints Tinder burder and the card moves from the sorted ist is returned. Intelled and burder and the card moves from the sorted ist is returned.	The backend prints an error message, and no new binder is added. Prints "Binde deleted," and the binder is removed from BinderManager. Prints "Deleted calleded," and the binder remains. Prints "Calledon canceled," and the binder remains. Prints "Card added," and the card moves from collection to binder. Prints "Faided to add card" and no state is changed. Prints "Gard returned to collection." and the card moves from binder to collection. Prints "Trade successful" and "Pikachu" is gone, "Charmander" is now in the binder. Prints "Trade successful" and "Pikachu" is gone, "Charmander" is now in the binder. Prints "Trade successful" and "Pikachu" and the trade is aborted. Prints "Trade ancieded." and no state is changed. Prints "Failed to add card or duplicate card)". State is unchanged. Prints "Card added." and the card moves from collection to deck. Prints "Card added and the card moves from collection to deck. Prints "Card removed and returned to collection." Prints "Card corecognating to the second item in the sorted list is returned. null is returned. Prints "Collection is empty" and returns null. The third Card object from the sorted list is returned.	P P P P P P P P P P P P P P P P P P P
handleDeleteBinder handleAddCardToBinder handleRemoveCardFromBin handleTradeFromBinder handleTradeFromBinder handleCateBebeck handleAddCardToBeck handleETendErfromDelethandleEtendErfromDelethandleEtendErfromDelethandleDeleteDeck	n	1 Successfully creates a new binder with a unique name. 2 Fails to create a binder with a unique name. 3 Successfully deletes an existing binder after confirmation. 2 Cannois deletion if user does not confirm. 1 Successfully adds an available count of a binder. 2 Fails to add a card with zero available count. 1 Successfully adds an available count. 1 Completes a valid trade. 2 Cannois an unfair trade upon user prompt. 3 Fails if nomoning card details are invalid. 1 Successfully creates a new deck. 2 Successfully adds a unique card to a deck. 2 Successfully adds a unique card to a deck. 3 Successfully removes a card from a deck. 4 Successfully removes a card from a deck. 5 Returns null when user chooses to go back. 6 Returns the correct Card object on valid selection. 7 Returns the correct Card object on valid selection. 8 Handles an empty collection gracefully. 8 Returns null when user chooses to go back. 9 Handles an empty collection gracefully.	User enters a valid, unique name "Trade Binder." User enters "Trade Binder," when it already exists. User selects "Trade Binder," then confirms with "yes". User selects "Trade Binder," then enters "no". User selects "Trade Binder," then selects "Pikachu" (count > 0). User selects "Trade Binder, then selects "Pikachu" (count > 0). User selects "Trade Binder, then selects "Alewhov" (count = 0). User selects "Trade Binder, then selects an existing card "Pikachu" from it. User selects a binder, selects "Pikachu" to trade away, enters valid details for a new card "Charmander". Trade results in value difference >= \$1.00. User enters "no" when prompted to proceed. User selects a binder, selects "Pikachu" to trade away, enters valid details for a new card "Charmander". Trade results in value difference >= \$1.00. User enters "no" when prompted to proceed. User selects "Alim Deck", then comming card. User selects "Main Deck", then selects "Squirtler" (which is not in the deck). User selects "Main Deck", then a card from its list. User selects "Alim Deck", then a card from its list. User selects and \$2 from the main collection list. User selects and \$2 from the main collection list. User selects is provided. User selects #3. A list of 3 cards is provided. A merply ArrayLst is provided.	Prints Timuter created successfully" and the binder is added to BinderManager. The backend prints a error message, and no new binder is added. Prints 'Binder deleted." and the binder is removed from BinderManager. Prints 'Deletion canceled." and the binder is removed from BinderManager. Prints 'Tealed canceled." and the binder remains. Prints 'Tarde and old card." and no state is changed. Prints 'Tarde acucessfull' and 'Pikachu' is gone. ("Dammander is now in the binder. Prints 'Trade canceled." and no state is changed. Prints 'Trade cand dad card. (or duplicate card)' State is undhanger. Prints Tarde and card (or duplicate card)' State is undhanged. Prints 'Tarde dadded." and the card moves from collection to deck. Prints 'Tard edeted." The Card object corresponding to the second item in the sorted list is returned. Prints 'Collection is empty" and returned until is returned. Prints 'Collection is empty" and returned until is returned. Prints 'Took on cards to select.' and returned until. Prints 'Took on cards to select.' and returned until.	The backend prints an error message, and no new binder is added. Prints "Bruder deleted", and the binder is removed from BinderManager. Prints "Deletion canceled." and the binder remains. Prints "Card added." and the prints "Ender on Collection to binder. Prints "Card added." and the card moves from collection to binder. Prints "Fade to add card." and no state is changed. Prints "Trade canceled." and no state is changed. Prints "Trade canceled." and no state is changed. Prints Trade deleted." and the card moves from collection to deck. Prints "Card deded." and the card moves from collection to deck. Prints Tock created successfully." and the deck is added to DeckManager. Prints "Card removed and returned to collection." Prints "Card removed and returned to collection." The trade disject corresponding to the second item in the sorted list is returned. Prints "Cald section is empty" and returns null. The third Card object from the sorted list is returned.	P P P P P P P P P P P P P P P P P P P
handleDeleteBinder handleAddCardToBinder handleAddCardToBinder handleTemoveCardFromBin handleTradeFromBinder handleCrateBoeck handleAddCardToDeck handleAddCardToDeck handleDeleteDeck selectCardFromCollection selectCardFromCollection	n	1 Successfully creates a new binder with a unique name. 2 Fails to create a binder with a unique name. 1 Successfully deletes an existing binder after confirmation. 2 Cancels deletion if user does not confirm. 1 Successfully adds an available card to a binder. 2 Fails to add a card with zero available count. 1 Successfully adds an available count of binder. 2 Fails to add a card with zero available count. 3 I Successfully adds an available card to a binder. 1 Completes a valid trade. 2 Cancels an unifer trade upon user prompt. 3 Fails if incoming card details are invalid. 1 Successfully reates a new deck. 1 Successfully reates an explicate card to a deck. 2 Successfully adds a unique card to a deck. 1 Successfully deletes a deck after confirmation. 1 Returns the corner Card object on valid selection. 2 Returns null when user chooses to go back. 3 Handles an empty collection gracefully. 1 Returns the cornect Card object or provided list. 2 Returns null if the provided list is empty. 1 Returns the correct Binder on valid selection.	User enters a valid, unique name "Trade Binder". User enters "Trade Binder" when it already exists. User selects "Trade Binder", then confirms with "yes". User selects "Trade Binder", then enters "no". User selects "Trade Binder", then selects "Rekarbu" (count > 0). User selects "Trade Binder", then selects "Rekarbu" (count > 0). User selects "Trade Binder", then selects "Rekarbu" (count > 0). User selects "Trade Binder", then selects are axisting card "Rikachu" from it. User selects a binder, selects "Rekarbu" to trade away, enters valid details for a new card "Charmander". Trade results in value difference >= \$1.00. User enters "no" when prompted to proceed. User enters a blank name for the incoming card. User enters a valid, unique name "Main Deck". User selects "Main Deck", then accord from the list. User selects "Main Deck", then a card from its list. User selects "Main Deck", then a card from its list. User selects "Main Deck", then a card from its list. User selects and #2 from the main collection list. User selects and #2 from the main collection list. User selects and \$2 from the main collection list. All st of 3 cards is provided. User selects #3. An empty Array, last is provided. User selects feet #1 from the list.	Prints "Binder created successfully." and the binder is added to BinderManager. The backend prints an error message, and no new binder is added. Prints "Binder detects" and the binder is removed from BinderManager. Prints "Deletion canceled." and the binder remains. Prints "Card added." and the card moves from collection to binder. Prints "Failed to add card" and no state is changed. Prints "Trade canceled." and search and the deck is aborted. Prints "Card dadd" and the card moves from collection to deck. Prints "Card removed and returned to collection." Prints "Card deck deleted." The Card object corresponding to the second item in the sorted list is returned. Prints "Card deck contrasponding to the second item in the sorted list is returned. Prints "There are no cards to select." and returnes null. The Binder object to the first item in the sorted list is returned.	The backend prints an error message, and no new binder is added. Prints "Binder deleted" and the binder is removed from BinderManager. Prints "Deletion canceled." and the binder remains. Prints "Card added." and the card moves from collection to binder. Prints "Failed to add card" and no state is changed. Prints "Failed to add card" and no state is changed. Prints "Trade canceled." and "Pikachu" is gone, "Charmander" is now in the binder. Prints "Trade canceled." and no state is changed. Prints "Trade canceled." and the cancel is added to DeckManager. Prints "Trade dead successfully" and the deck is added to DeckManager. Prints "Trade add card or duplicate card)". State is unchanged. Prints "Card dead" and the card moves from collection to deck. Prints "Card aremoved and returned to collection." Prints "Deck deleted." The Card chipect corresponding to the second item in the sorted list is returned. Prints "Card collection is empty" and returns null. The third Card object from the sorted list is returned. Prints "There are no cards to select." and returns null.	P P P P P P P P P P P P P P P P P P P
handleDeleteBinder handleAddCardToBinder handleAddCardToBinder handleTradeFromBinder handleTradeFromBinder handleCteateDeck handleAddCardToDeck handleRemoveCardFromDes handleDeleteDeck	n	1 Successfully creates a new binder with a unique name. 2 Fails to create a binder with a unique name. 3 Successfully deletes an existing binder after confirmation. 2 Cannois deletion if user does not confirm. 1 Successfully adds an available count of a binder. 2 Fails to add a card with zero available count. 1 Successfully adds an available count. 1 Completes a valid trade. 2 Cannois an unfair trade upon user prompt. 3 Fails if nomoning card details are invalid. 1 Successfully creates a new deck. 2 Successfully adds a unique card to a deck. 2 Successfully adds a unique card to a deck. 3 Successfully removes a card from a deck. 4 Successfully removes a card from a deck. 5 Returns null when user chooses to go back. 6 Returns the correct Card object on valid selection. 7 Returns the correct Card object on valid selection. 8 Handles an empty collection gracefully. 8 Returns null when user chooses to go back. 9 Handles an empty collection gracefully.	User enters a valid, unique name "Trade Binder." User enters "Trade Binder," when it already exists. User selects "Trade Binder," then confirms with "yes". User selects "Trade Binder," then enters "no". User selects "Trade Binder," then selects "Pikachu" (count > 0). User selects "Trade Binder, then selects "Pikachu" (count > 0). User selects "Trade Binder, then selects "Alewhov" (count = 0). User selects "Trade Binder, then selects an existing card "Pikachu" from it. User selects a binder, selects "Pikachu" to trade away, enters valid details for a new card "Charmander". Trade results in value difference >= \$1.00. User enters "no" when prompted to proceed. User selects a binder, selects "Pikachu" to trade away, enters valid details for a new card "Charmander". Trade results in value difference >= \$1.00. User enters "no" when prompted to proceed. User selects "Alim Deck", then comming card. User selects "Main Deck", then selects "Squirtler" (which is not in the deck). User selects "Main Deck", then a card from its list. User selects "Alim Deck", then a card from its list. User selects and \$2 from the main collection list. User selects and \$2 from the main collection list. User selects is provided. User selects #3. A list of 3 cards is provided. A merply ArrayLst is provided.	Prints Timuter created successfully" and the binder is added to BinderManager. The backend prints a error message, and no new binder is added. Prints 'Binder deleted." and the binder is removed from BinderManager. Prints 'Deletion canceled." and the binder is removed from BinderManager. Prints 'Tealed canceled." and the binder remains. Prints 'Tarde and old card." and no state is changed. Prints 'Tarde acucessfull' and 'Pikachu' is gone. ("Dammander is now in the binder. Prints 'Trade canceled." and no state is changed. Prints 'Trade cand dad card. (or duplicate card)' State is undhanger. Prints Tarde and card (or duplicate card)' State is undhanged. Prints 'Tarde dadded." and the card moves from collection to deck. Prints 'Tard edeted." The Card object corresponding to the second item in the sorted list is returned. Prints 'Collection is empty" and returned until is returned. Prints 'Collection is empty" and returned until is returned. Prints 'Took on cards to select.' and returned until. Prints 'Took on cards to select.' and returned until.	The backend prints an error message, and no new binder is added. Prints "Bruder deleted", and the binder is removed from BinderManager. Prints "Deletion canceled." and the binder remains. Prints "Card added." and the prints "Ender on Collection to binder. Prints "Card added." and the card moves from collection to binder. Prints "Fade to add card." and no state is changed. Prints "Trade canceled." and no state is changed. Prints "Trade canceled." and no state is changed. Prints Trade deleted." and the card moves from collection to deck. Prints "Card deded." and the card moves from collection to deck. Prints Tock created successfully." and the deck is added to DeckManager. Prints "Card removed and returned to collection." Prints "Card removed and returned to collection." The trade disject corresponding to the second item in the sorted list is returned. Prints "Cald section is empty" and returns null. The third Card object from the sorted list is returned.	P P P P P P P P P P P P P P P P P P P

		1 Returns the correct Deck on valid selection.	User selects deck #2 from the list.	The Deck object for the second item in the sorted list is returned.	The Deck object for the second item in the sorted list is returned.	P
selectDeck		2 Returns null when no decks exist.	Method is called when deck list is empty.	Prints "No decks exist." and returns null.	Prints "No decks exist." and returns null.	P
		1 Returns the correct index for an existing card.	List contains ["A", "B", "C"]. Search for "B".	1		1 P
findCardIndexInList		2 Returns -1 for a non-existent card.	List contains ["A", "B", "C"]. Search for "D".	-1		-1 P
		•	Class: Menu			
Method	#	Test Description	Sample Input Data	Expected Output	Actual Output	P/F
		Loops and waits for input, correctly delegates to a sub-menu.	User enters "1".	The Collection Menu is displayed.	The Collection Menu is displayed.	P
runMainMenu		2 Exits the loop when user enters "0".	User enters "0".	The runMainMenu method finishes execution.	The runMainMenu method finishes execution.	P
runCollectionMenu		Loops and correctly delegates to a handler method.	User enters "2".	The handler.handleAddNewCard() method is called.	The handler.handleAddNewCard() method is called.	P
runBinderMenu		Loops and correctly delegates to a handler method.	User enters "2".	The handler.handleCreateBinder() method is called.	The handler.handleCreateBinder() method is called.	P
runDeckMenu		Loops and correctly delegates to a handler method.	User enters "2".	The handler.handleCreateDeck() method is called.	The handler.handleCreateDeck() method is called.	P