| MCO1 Test Script | SUBMITTED BY: | | Edriene James Paingan [12413984] | Franz Patrick Magbitang [12414409] | | | |
|---|---|---|---|---|---|---|---|
| | | | | | | | |
| **Class** | **Method** | **#** | **Description** | **Sample Input Data** | **Expected Output** | **Actual Output** | **P/F** |
| | | | | **Models** | | | |
| **Collection** | Collection() | 1 | Constructor | N/A (Initialization) | New Collection instance | Same | P |
| | | | | | | | |
| | addCard(String cardName, Rarity rarity, Variant variant, Double value) | 1 | Add a valid common card | "Fireball", Rarity.COMMON, Variant.NORMAL, 10.0 | Collection contains "Fireball" with value $10.00 | "Fireball" added | P |
| | | 2 | Add a rare card with a special variant | "Ice Dragon", Rarity.RARE, Variant.ALT_ART, 50.0 | Collection contains "Ice Dragon" with value 50.0 * multiplier | "Ice Dragon" added | P |
| | | 3 | Add a second card and verify alphabetical sorting | After adding "Fireball", add "Arcane Blast" with any valid attributes | List is sorted: "Arcane Blast" appears before "Fireball" | Sorted correctly | P |
| | | | | | | | |
| | addCard(Card card) | 1 | Add a new Card object to the collection | Card("Wind Spirit", Rarity.UNCOMMON, Variant. NORMAL, 15.0) | Collection contains "Wind Spirit" | "Wind Spirit" added | P |
| | | 2 | Add multiple Card objects and check sort order | Add: Card("Boulder", Rarity.COMMON, ...), then Card("Abyss Fiend", Rarity.RARE, ...) | Collection sorted alphabetically: "Abyss Fiend" before "Boulder" | Sorted correctly | P |
| | | 3 | Add duplicate card names to test sorting stability | Add: Card("Phantom", Rarity.LEGENDARY, Variant. ALT_ART, 100.0), then another "Phantom" with same name | Both cards appear (duplicates allowed), order based on insertion | Duplicates handled | P |
| | | | | | | | |
| | displayCollection() | 1 | Display empty collection | No cards added | "Collection is empty" | Printed as expected | P |
| | | 2 | Display collection with multiple cards with count > 0 | Add Card("Dragon", COMMON, NORMAL, 10.0)<br>Add Card("Elf", RARE, FULL_ART, 20.0) | Lists "Dragon" and "Elf" with their counts | Printed as expected | P |
| | | 3 | Display collection with some cards having count = 0 | Add Card("Goblin", COMMON, NORMAL, 5.0) with count 0<br>Add Card("Knight", RARE, ALT_ART, 25.0) with count 2 | Only "Knight" is shown, "Goblin" is omitted | Printed as expected | P |
| | | | | | | | |
| | displayCard() | 1 | Display a card that exists and has count > 0 | Add card "Phoenix" with count 1<br>Call displayCard("Phoenix") | Card details printed: name = "Phoenix", rarity = correct, value shown<br>Returns true | Printed + returns true | P |
| | | 2 | Try to display a card that exists but count is 0 | Add card "Zombie" with count 0<br>Call displayCard("Zombie") | No output printed, returns false | Nothing printed, returns false | P |
| | | 3 | Try to display a card that does not exist in collection | Call displayCard("Ghost") without adding such card | No output printed, returns false | Nothing printed, returns false | P |
| | | | | | | | |
| | changeCardCount(String name, int count) | 1 | Increase count by a positive number | Add "Blue-Eyes" with count 1<br>Call increaseCardCount("Blue-Eyes", 2) | Count becomes 3<br>Prints success | Count = 3<br>Increment Successful! | P |
| | | 2 | Try to increase count for card that doesn't exist | Call increaseCardCount("Dark Magician", 5) without adding it | Print "Card not found." | "Card not found." printed | P |
| | | 3 | Increase count using a negative number (decrease) | Add "Red-Eyes" with count 5<br>Call increaseCardCount("Red-Eyes", -2) | Count becomes 3<br>Prints success | Count = 3<br>Increment Successful! | P |
| | | | | | | | |
| | searchCard(String name) | 1 | Successfully finds an existing card | Add "Pikachu" to collection<br>Search for "Pikachu" | Returns Card with name "Pikachu" | Returns Card with name "Pikachu" | P |
| | | 2 | Case-insensitive search works correctly | Add "Charizard" to collection<br>Search for "charizard" (lowercase) | Returns Card with name "Charizard" | Returns Card with name "Charizard" | P |
| | | 3 | Card not found in collection | No cards added<br>Search for "Snorlax" | Returns null | Returns null | |
| | | | | | | | |
| | isEmpty() | 1 | Empty collection — no cards added | No cards in collection | TRUE | TRUE | P |
| | | 2 | All cards have 0 count | Add "Bulbasaur" with count 0 | TRUE | TRUE | P |
| | | 3 | At least one card has count > 0 | Add "Squirtle" with count 1 | FALSE | FALSE | P |
| | **Getters and Setters** | | | | | | |
| | getAllCards() | 1 | Returns all cards | Function Call | Returns all cards | Returns all cards | P |
| | | | | | | | |
| **Card** | Card(String name, Rarity rarity, Variant variant, double baseValue) | 1 | Create a normal common card | "Pikachu", Rarity.COMMON, Variant.NORMAL, 10.0 | actualValue = 10.0, count = 1 | actualValue = 10.0, count = 1 | P |
| | | 2 | Create a rare full-art card | "Charizard", Rarity.RARE, Variant.FULL_ART, 20.0 | actualValue = 20.0 * FULL_ART multiplier, count = 1 | actualValue = X, count = 1 | P |
| | | 3 | Increase count from 1 to 6 | Call incrementCount(5) on a card created with default count 1 | count = 6 | count = 6 | P |
| | | | | | | | |
| | incrementCount(int count) | 1 | Increment by positive number | initialCount = 1, call incrementCount(3) | count = 4 | count = 4 | P |
| | | 2 | Decrement by 1 (reduce count safely) | initialCount = 5, call incrementCount(-1) | count = 4 | count = 4 | P |
| | | 3 | Increment by 0 (no change) | initialCount = 2, call incrementCount(0) | count = 2 | count = 2 | P |
| | | | | | | | |
| | viewCardDetails() | 1 | Displays a common, normal card | name = "Pikachu", rarity = COMMON, variant = NORMAL, baseValue = 1.00 | Pikachu COMMON NORMAL $1.00 | Pikachu COMMON NORMAL $1.00 | P |
| | | 2 | Displays a rare, full art card | name = "Charizard", rarity = RARE, variant = FULL_ART, baseValue = 5.00 | Charizard RARE FULL_ART $7.50 | Charizard RARE FULL_ART $7.50 | P |

| MCO1 Test Script | SUBMITTED BY: | | Edriene James Paingan [12413984] | Franz Patrick Magbitang [12414409] | | | |
|---|---|---|---|---|---|---|---|
| **Class** | **Method** | **#** | **Description** | **Sample Input Data** | **Expected Output** | **Actual Output** | **P/F** |
| | | 3 | Displays a legendary, alt art card | name = "Dark Magician", rarity = LEGENDARY, variant = ALT_ART, baseValue = 10 | Dark Magician LEGENDARY ALT_ART $20.00 | Dark Magician LEGENDARY ALT_ART $20.00 | P |
| | **Getters and Setters** | | | | | | |
| | getName() | 1 | Returns card name | Function Call | Returns card name | Returns card name | P |
| | getCount() | 1 | Returns card count | Function Call | Returns card count | Returns card count | P |
| | getActualValue() | 1 | Returns card's actual value | Function Call | Returns card's actual value | Returns card's actual value | P |
| **Binder** | Binder(String name) | 1 | Create binder with a simple valid name | "MyBinder" | Binder name = "MyBinder", cards = empty list | Binder name = "MyBinder", cards = empty list | P |
| | | 2 | Create binder with special characters in name | "Rare Binder#1" | Binder name = "Rare Binder#1", cards = empty list | Binder name = "Rare Binder#1", cards = empty list | P |
| | | 3 | Create binder with empty string as name | "" | Binder name = "", cards = empty list | Binder name = "", cards = empty list | P |
| | addCard(Card card) | 1 | Add a single valid card to an empty collection | Card: "Pikachu", Rarity: COMMON, Variant: NORMAL | Collection contains 1 card: "Pikachu" | Collection contains 1 card: "Pikachu" | P |
| | | 2 | Add two cards and check alphabetical sorting | Card1: "Zubat", Card2: "Abra" | Collection is sorted as: "Abra", "Zubat" | Collection is sorted as: "Abra", "Zubat" | P |
| | | 3 | Add duplicate name card and check list growth | Two cards with name: "Charmander" | Collection size = 2 (duplicates allowed), both named "Charmander" | Collection size = 2, both named "Charmander" | P |
| | removeCard(Card card) | 1 | Remove an existing card from a binder | Binder contains: "Pikachu"<br>Remove: "Pikachu" | Binder no longer contains "Pikachu" | Binder no longer contains "Pikachu" | P |
| | | 2 | Attempt to remove a card that does not exist | Binder contains: "Pikachu"<br>Remove: "Charmander" | Binder remains unchanged | Binder remains unchanged | P |
| | | 3 | Remove a card from a binder with multiple cards | Binder contains: "Zubat", "Abra", "Pikachu"<br>Remove: "Abra" | Binder contains: "Zubat", "Pikachu" | Binder contains: "Zubat", "Pikachu" | P |
| | isFull() | 1 | Empty binder should not be full | Binder has 0 cards | FALSE | FALSE | P |
| | | 2 | Binder with exactly 20 cards is full | Binder has 20 cards (added manually) | TRUE | TRUE | P |
| | | 3 | Binder with 21 cards is still full | Binder has 21 cards | TRUE | TRUE | P |
| | viewBinder() | 1 | Empty binder should print empty message | Binder name: "MyBinder", no cards | Prints banner + "THIS BINDER IS EMPTY" | As expected | P |
| | | 2 | Binder with one card should show its details | Binder has 1 card: Pikachu, RARE, FULL_ART, $10.00 | Prints header + one line: Pikachu, RARE, FULL_ART, $10.00 | As expected | P |
| | | 3 | Binder with multiple cards shows them all | Binder has 3 cards: Pikachu, Charizard, Bulbasaur with valid details | Prints all 3 card details in table format sorted alphabetically by name | As expected | P |
| | searchCard(String name) | 1 | Card found (exact match) | Cards in binder: Pikachu, Charizard<br>Input: "Pikachu" | Returns Card object with name "Pikachu" | As expected | P |
| | | 2 | Card found (case-insensitive match) | Cards in binder: Pikachu, Charizard<br>Input: "charizard" | Returns Card object with name "Charizard" | As expected | P |
| | | 3 | Card not found | Cards in binder: Pikachu, Charizard<br>Input: "Bulbasaur" | Returns null | As expected | P |
| | **Getters and Setters** | | | | | | |
| | getName() | 1 | Returns binder name | Function Call | Returns binder name | Returns binder name | P |
| | getCards() | 1 | Returns list of cards in binder | Function Call | Returns list of cards in binder | Returns list of cards in binder | P |
| **Deck** | Deck(String name) | 1 | Create deck with valid name | "My Deck" | New deck with name "My Deck" and empty card list | Same | P |
| | | 2 | Create deck with empty string | "" | Deck created with empty name and empty card list | Same | P |
| | | 3 | Create deck with null (invalid input) | null | Nothing happens | Same | P |
| | addCard(Card card) | 1 | Add card to empty deck | Card("Dragon") | Card "Dragon" added | Same | P |
| | | 2 | Add multiple cards | Card("Goblin"), Card("Elf") | All cards added in order | Same | P |
| | | 3 | Add null card (invalid input) | null | Nothing happens | Same | P |
| | removeCard(Card card) | 1 | Remove card that exists | Card("Goblin") in deck | Card removed successfully | Same | P |
| | | 2 | Remove card that doesn't exist | Card("Phoenix") not in deck | No effect or error | No change | P |
| | | 3 | Remove from empty deck | Card("Any") | No effect | No change | P |
| | isFull() | 1 | Deck has exactly 20 cards | 20 cards added | TRUE | TRUE | P |

| MCO1 Test Script | SUBMITTED BY: | | Edriene James Paingan [12413984] | Franz Patrick Magbitang [12414409] | | | |
|---|---|---|---|---|---|---|---|
| | | | | | | | |
| **Class** | **Method** | **#** | **Description** | **Sample Input Data** | **Expected Output** | **Actual Output** | **P/F** |
| | | 2 | Deck has less than 20 cards | 5 cards added | FALSE | FALSE | P |
| | | 3 | Deck has more than 20 cards | 21 cards added | TRUE | TRUE | P |
| | | | | | | | |
| | viewDeck() | 1 | View empty deck | No cards | Message: "Deck is empty" | Same | P |
| | | 2 | View deck with one card | Card("Dragon") | Displays Dragon's name | Same | P |
| | | 3 | View deck with multiple cards | 3 cards | All names printed line by line | Same | P |
| | | | | | | | |
| | displayCard(String cardName) | 1 | Display existing card | "Dragon" | Card details printed, returns true | Same | P |
| | | 2 | Display card not in deck | "Unicorn" | Returns false, no print | Same | P |
| | | 3 | Display with different case | "gObLiN" (in deck as "Goblin") | Case-insensitive match, returns true | Same | P |
| | | | | | | | |
| | searchCard(String name) | 1 | Search existing card | "Dragon" | Card("Dragon") | Same | P |
| | | 2 | Search non-existing card | "Zombie" | null | Same | P |
| | | 3 | Case-insensitive search | "eLf" (deck has "Elf") | Card("Elf") | Same | P |
| | **Getters and Setters** | | | | | | |
| | getName() | 1 | Returns deck name | Function Call | Returns deck name | Returns deck name | P |
| | | | | | | | |
| | getCards() | 1 | Returns list of cards in deck | Function Call | Returns list of cards in deck | Returns list of cards in deck | P |
| | | | | **Managers** | | | |
| **ManageBinders** | ManageBinders(Collection collection) | 1 | Initialize with valid collection | Collection with cards | binders = [], collection linked, tradeCard initialized | Same | P |
| | | 2 | Initialize with empty collection | Empty Collection | No exception, binders = [] | Same | P |
| | | 3 | Initialize with null | null | Nothing Happens | Same | P |
| | | | | | | | |
| | createBinder(String name) | 1 | Create new binder | "MyBinder" | Binder added | Same | P |
| | | 2 | Create with duplicate name | "MyBinder" again | Another binder still added | Same | P |
| | | 3 | Create with empty name | "" | Empty-name binder added | Same | P |
| | | | | | | | |
| | deleteBinder(String binderName) | 1 | Delete existing binder | "MyBinder" | true, binder removed | Same | P |
| | | 2 | Delete non-existing binder | "UnknownBinder" | FALSE | Same | P |
| | | 3 | Delete binder and restore cards | "FilledBinder" | Cards' counts incremented | Same | P |
| | | | | | | | |
| | addCardToBinder(String cardName, String binderName) | 1 | Valid transfer | "Elf", "MyBinder" | TRUE | Same | P |
| | | 2 | Binder full | Card count > 0, binder full | FALSE | Same | P |
| | | 3 | Card not found in collection | "Ghost", "MyBinder" | FALSE | Same | P |
| | | | | | | | |
| | removeCardFromBinder(String cardName, String binderName) | 1 | Valid removal | "Elf", "MyBinder" | true, count++ | Same | P |
| | | 2 | Card not in binder | "Dragon", "MyBinder" | FALSE | Same | P |
| | | 3 | Binder not found | "Elf", "UnknownBinder" | FALSE | Same | P |
| | | | | | | | |
| | tradeCard(String cardName, String binderName) | 1 | Successful trade (incoming != null) | "Elf", "MyBinder" | Card replaced in collection and binder | Same | P |
| | | 2 | Trade rejected (incoming == null) | "Dragon", "MyBinder" | Same card returned to binder | Same | P |
| | | 3 | Invalid card input | "FakeCard", "MyBinder" | No changes made | Same | P |
| | | | | | | | |
| | viewSpecificBinder(String binderName) | 1 | View valid binder | "MyBinder" | Prints binder contents | Same | P |
| | | 2 | View non-existent binder | "NoBinder" | false, nothing printed | Same | P |
| | | 3 | View empty binder | "EmptyBinder" | "Binder is empty" message | Same | P |
| | | | | | | | |
| | searchBinder(String name) | 1 | Search existing binder | "MyBinder" | Binder object found | Same | P |
| | | 2 | Case-insensitive search | "mybinder" | Binder object found | Same | P |
| | | 3 | Search non-existent binder | "GhostBinder" | null | Same | P |
| | | | | | | | |
| **ManageDeck** | ManageDeck(Collection collection) | 1 | Initialize with non-empty collection | Collection w/ 5 cards | Empty deck list, collection is linked | Same | P |
| | | 2 | Initialize with empty collection | Empty Collection | No decks, no errors | Same | P |
| | | 3 | Initialize with null (if allowed) | null | NullPointerException or handled | Handled | P |
| | | | | | | | |
| | createDeck(String name) | 1 | Create new deck | "MyDeck" | Deck added to list | Same | P |
| | | 2 | Create multiple decks | "DeckA", "DeckB" | Both added | Same | P |
| | | 3 | Create with duplicate name | "MyDeck" again | Both exist (no check for duplicates) | Same | P |
| | | | | | | | |
| | deleteDeck(String deckName) | 1 | Delete existing deck | "MyDeck" | Deck removed, returns true | Same | P |
| | | 2 | Delete non-existent deck | "FakeDeck" | Returns false | Same | P |
| | | 3 | Delete deck and restore card counts | "DeckWithCards" | All cards' count incremented | Same | P |

| MCO1 Test Script | SUBMITTED BY: | | Edriene James Paingan [12413984] | Franz Patrick Magbitang [12414409] | | | |
|---|---|---|---|---|---|---|---|
| | | | | | | | |
| **Class** | **Method** | **#** | **Description** | **Sample Input Data** | **Expected Output** | **Actual Output** | **P/F** |
| | | | | | | | |
| | searchDeck(String name) | 1 | Search existing deck | "MyDeck" | Deck object found | Same | P |
| | | 2 | Case-insensitive search | "mYdEcK" | Deck object found | Same | P |
| | | 3 | Search non-existent deck | "UnknownDeck" | null | Same | P |
| | | | | | | | |
| | addCardToDeck(String cardName, String deckName) | 1 | CC | "Elf", "MyDeck" | TRUE | Same | P |
| | | 2 | Card not found in collection | "Ghost", "MyDeck" | FALSE | Same | P |
| | | 3 | Deck full | Deck already has 20 cards | FALSE | Same | P |
| | | | | | | | |
| | removeCardFromDeck(String cardName, String deckName) | 1 | Remove existing card | "Elf", "MyDeck" | true, count++ | Same | P |
| | | 2 | Card not in deck | "Dragon", "MyDeck" | FALSE | Same | P |
| | | 3 | Invalid deck name | "Elf", "NoDeck" | FALSE | Same | P |
| | | | | | | | |
| | viewSpecificDeck(String deckName) | 1 | Card exists in deck | "Elf", Deck with "Elf" | Card info printed, true | Same | P |
| | | 2 | Card not in deck | "Goblin", Deck w/o Goblin | FALSE | Same | P |
| | | 3 | Case-insensitive match | "eLf", deck has "Elf" | TRUE | Same | P |
| | | | | | | | |
| | viewSpecificCardinDeck(String cardName, Deck deck) | 1 | Card exists in deck | "Elf", Deck with "Elf" | Card info printed, true | Same | P |
| | | 2 | Card not in deck | "Goblin", Deck w/o Goblin | FALSE | Same | P |
| | | 3 | Case-insensitive match | "eLf", deck has "Elf" | TRUE | Same | P |
| **Menus** | | | | | | | |
| **Menu** | Menu() | 1 | Initializes all subsystems correctly | N/A | collection, binderUI, deckUI, etc. are non-null | All non-null | P |
| | | 2 | Controllers are linked with shared scanner | N/A | All controllers use same scanner | Shared scanner | P |
| | | 3 | Uses Collection as shared reference | N/A | manageBinder, manageDeck, and collectionUI use same Collection | Confirmed | P |
| | | | | | | | |
| | run() | 1 | Valid input: 1 | "1\n" | 1 | 1 | P |
| | | 2 | Valid input: 4 (exit) | "4\n" | 4 | 4 | P |
| | | 3 | Input buffering check (follow-up input) | "3\n" | 3 | 3 | P |
| | | | | | | | |
| | displayMainMenu() | 1 | Select Collection → exit | "1\n4\n" | collectionUI.collectionMenu() called | Called | P |
| | | 2 | Select Decks → exit | "3\n4\n" | deckUI.manageDeckMenu() called | Called | P |
| | | 3 | Exit immediately | "4\n" | System exits with no module interaction | Loop ends | P |
| | | | | | | | |
| **CollectionController** | CollectionController(Collection collection, Scanner scanner) | 1 | Initializes with valid arguments | mockCollection, mockScanner | Non-null internal fields | Non-null | P |
| | | 2 | Scanner is stored properly | new Scanner(...) | Can access and use scanner | Yes | P |
| | | 3 | Collection is stored properly | mockCollection | Can call collection methods | Yes | P |
| | | | | | | | |
| | collectionMenu() | 1 | Valid input (3) | "3\n" | 3 | 3 | P |
| | | 2 | Invalid input (non-integer) | "abc\n" | -1 and message | -1 | P |
| | | 3 | Valid input (4 = exit) | "4\n" | 4 | 4 | P |
| | | | | | | | |
| | collectionMenuTemplate() | 1 | Select add card then exit | "1\nCardX\nrare\nnormal\n5.0\n4\n" | Card added, then program exits | Matches expected | P |
| | | 2 | Choose display collection then exit | "3\n2\n4\n" | Collection displayed then exit | Matches expected | P |
| | | 3 | Choose increase, enter card, then exit | "2\n1\nCardY\n3\n4\n" | CardY count increased, then exited | Matches expected | P |
| | | | | | | | |
| | increaseDecrease() | 1 | Increase existing card count | "1\nCardA\n3\n" | increaseCardCount("CardA") called | Called | P |
| | | 2 | Decrease existing card count | "2\nCardB\n3\n" | decreaseCardCount("CardB") called | Called | P |
| | | 3 | Exit immediately | "3\n" | No card modified | No effect | P |
| | | | | | | | |
| | display() | 1 | View a known card | "1\nMyCard\n3\n" | Card details shown | Correct display | P |
| | | 2 | View full collection | "2\n3\n" | All cards shown | Full display | P |
| | | 3 | Invalid card name (not found) | "1\nGhostCard\n3\n" | "Card not found" message | Matches expected | P |
| | | | | | | | |
| | addInputCard() | 1 | Add brand new rare card with variant | "CardZ\nrare\nfull_art\n12.5\n" | addCard(...) with correct data | Called | P |
| | | 2 | Try to add duplicate card, say yes | "CardX\nY\n" | incrementCount() on existing card | Called | P |
| | | 3 | Invalid rarity entered twice | "CardA\nnot_a_rarity\nrare\nnormal\n8.0\n" | Retry until valid rarity, then add | Prompts, then add | P |
| | | | | | | | |
| **BindersController** | BindersController(ManageBinders manageBinder, Scanner scanner) | 1 | Controller initializes without error | Mock ManageBinders, Scanner | Controller instance created | Created | P |
| | | 2 | Scanner is stored | new Scanner(System.in) | Controller uses passed scanner | Not null | P |
| | | 3 | ManageBinders reference is stored | mockManageBinder | Internal manageBinder is accessible | Not null | P |
| | | | | | | | |
| | manageBinderMenuTemplate() | 1 | Choose option 1 (create binder) | "1\n" | 1 | 1 | P |

| MCO1 Test Script | SUBMITTED BY: | | Edriene James Paingan [12413984] | Franz Patrick Magbitang [12414409] | | | |
|---|---|---|---|---|---|---|---|
| | | | | | | | |
| **Class** | **Method** | **#** | **Description** | **Sample Input Data** | **Expected Output** | **Actual Output** | **P/F** |
| | | 2 | Choose option 7 (exit) | "7\n" | 7 | 7 | P |
| | | 3 | Invalid input | "hello\n" | Exception | Exception | P |
| | | | | | | | |
| | manageBinderMenu() | 1 | Create binder then exit | "1\nBinderX\n7\n" | BinderX created | Output matches | P |
| | | 2 | Delete nonexistent binder | "2\nGhostBinder\n7\n" | "Attempt to delete Binder Failed" | Matches expected | P |
| | | 3 | Trade card from existing binder | "5\nBinderY\nCardZ\n7\n" | TradeCard method triggered | Output as expected | P |
| | | | | | | | |
| | addCardToBinder() | 1 | Add valid card to valid binder | "CardA\nBinderB\n" | Success message: CardA added to BinderB | Matches expected | P |
| | | 2 | Add card to non-existent binder | "CardX\nBinderGhost\n" | "Failed to add card to binder" | Matches expected | P |
| | | 3 | Add non-existent card | "GhostCard\nBinderB\n" | "Failed to add card to binder" | Matches expected | P |
| | | | | | | | |
| | removeCardFromBinder() | 1 | Remove card that exists in binder | "CardZ\nBinderY\n" | Success message: CardZ removed from BinderY | Matches expected | P |
| | | 2 | Remove card not in binder | "UnknownCard\nBinderY\n" | "Failed to remove card from binder" | Matches expected | P |
| | | 3 | Remove from non-existent binder | "CardX\nGhostBinder\n" | "Failed to remove card from binder" | Matches expected | P |
| | | | | | | | |
| **TradeCardController** | TradeCardController(Collection collection) | 1 | Initializes controller with collection | MockCollection | scanner and collection are non-null | Non-null | P |
| | | 2 | Scanner is initialized automatically | MockCollection | scanner != null | Pass | P |
| | | 3 | Stores reference to collection | MockCollection with stubbed card data | Collection is accessible by controller | Access OK | P |
| | | | | | | | |
| | tradeCardMenu(String cardName) | 1 | Trade a new card with valid rarity/variant | "NewCard\nrare\nfull_art\n10\n1\n" | Returns new Card("NewCard", ...) | Matches expected | P |
| | | 2 | Cancel trade after seeing value comparison | "NewCard\nrare\nfull_art\n10\n2\n" | null | null | P |
| | | 3 | Card already exists, accept count increment | "OldCard\nY\n" | No new card returned | null (count inc) | P |
| | | | | | | | |
| | displayTradeMenu(String cardName) | 1 | Fair trade: value difference is less than $1 | incomingCard: {name="Blue-Eyes", value=10.00}<br>outgoingCard: {name="Dark Magician", value=9.50}<br>User Input: 1 | Shows both cards, no “Value difference” message<br>Asks for trade confirmation<br>Returns "1" | "1" | P |
| | | 2 | Unfair trade: value difference is greater than $1 | incomingCard: {name="Charizard", value=15.00}<br>outgoingCard: {name="Pikachu", value=10.00}<br>User Input: 0 | Shows both cards, prints “Value difference is more than or equal to $1.”<br>Asks for trade confirmation<br>Returns "0" | "0" | P |
| | | 3 | Edge case: value difference is exactly $1 (should be unfair) | incomingCard: {name="Zebra", value=8.00}<br>outgoingCard: {name="Lion", value=7.00}<br>User Input: 1 | Shows both cards, prints “Value difference is more than or equal to $1.”<br>Asks for trade confirmation<br>Returns "1" | "1" | P |
| | | | | | | | |
| **DeckController** | DeckController(ManageDeck manageDeck, Scanner scanner) | 1 | Create controller with mock objects | mockManageDeck, mockScanner | Controller initializes without error | No exception | P |
| | | 2 | Ensure scanner is stored | new ManageDeck(), new Scanner(System.in) | Internal scanner is not null | Not null | P |
| | | 3 | Ensure manageDeck is stored | new ManageDeck(), new Scanner(...) | Internal manageDeck is not null | Not null | P |
| | | | | | | | |
| | manageDeckMenuTemplate() | 1 | Valid user input: "1" | "1\n" | 1 | 1 | P |
| | | 2 | Valid user input: "6" (exit option) | "6\n" | 6 | 6 | P |
| | | 3 | Invalid input (non-integer) | "hello\n" | Exception | Exception | P |
| | | | | | | | |
| | manageDeckMenu() | 1 | Create and view deck, then exit | "1\nMyDeck\n5\nMyDeck\nSomeCard\n6\n" | Deck created and card not found in view | Matches expected | P |
| | | 2 | Add and remove card from deck | "3\nCardX\nDeckY\n4\nCardX\nDeckY\n6\n" | Card added, then removed (with status print) | Matches expected | P |
| | | 3 | Try deleting non-existing deck | "2\nNonExistentDeck\n6\n" | "Deck deletion failed" | Matches expected | P |
| | | | | **Enums** | | | |
| **Variant** | getMultiplier() | 1 | Returns multiplier for NORMAL variant | Variant.NORMAL | 1 | 1 | P |
| | | 2 | Returns multiplier for EXTENDED_ART | Variant.EXTENDED_ART | 1.5 | 1.5 | P |
| | | 3 | Returns multiplier for FULL_ART variant | Variant.FULL_ART | 2 | 2 | P |
| | | 4 | Returns multiplier for ALT_ART variant | Variant.ALT_ART | 3 | 3 | P |