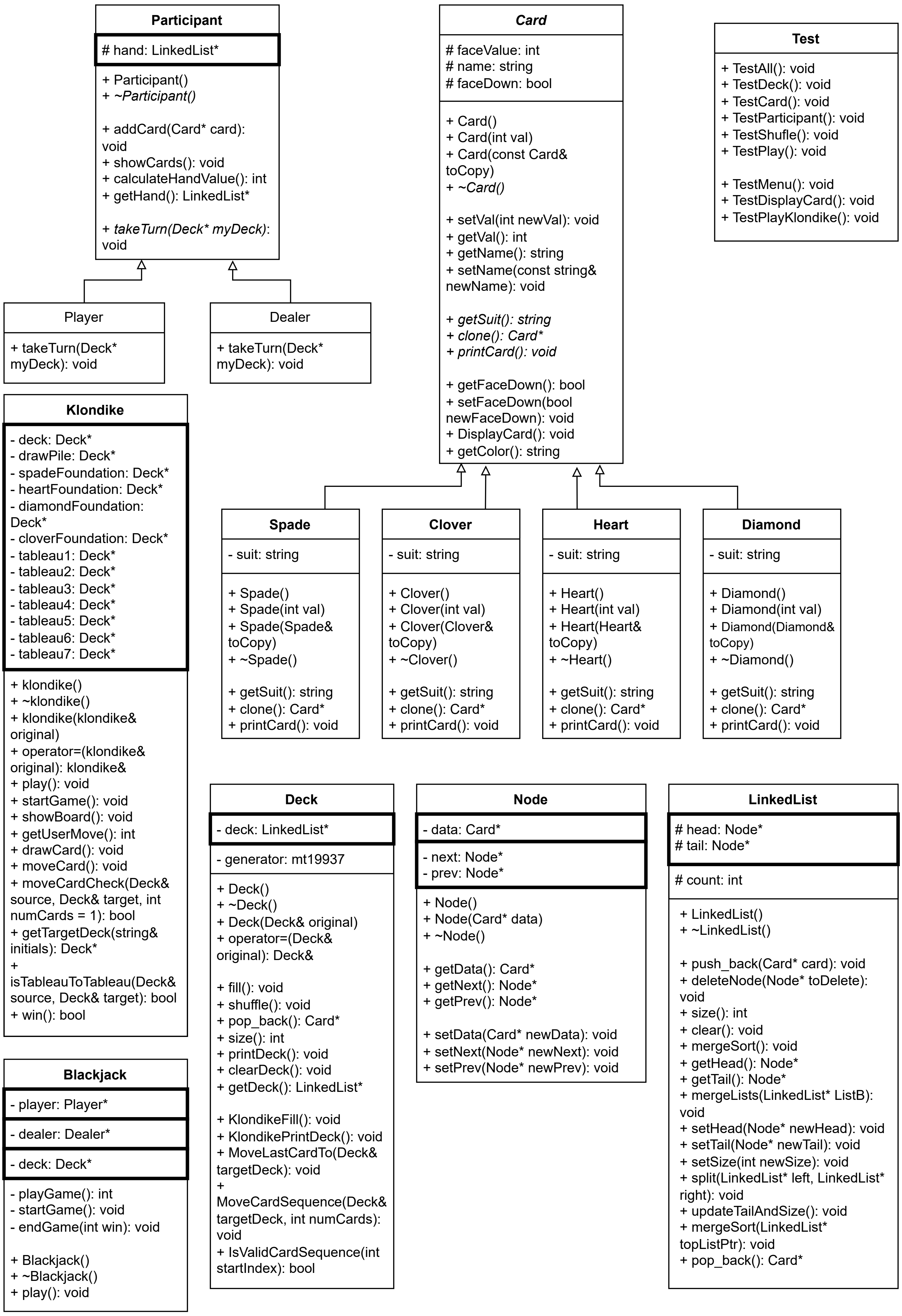

```
Participant
---
# hand: LinkedList*
---
+ Participant()
+ ~Participant()

+ addCard(Card* card): void
+ showCards(): void
+ calculateHandValue(): int
+ getHand(): LinkedList*

+ takeTurn(Deck* myDeck): void
```

```
Player
---
+ takeTurn(Deck* myDeck): void
```

```
Dealer
---
+ takeTurn(Deck* myDeck): void
```

```
Card
---
# faceValue: int
# name: string
# faceDown: bool
---
+ Card()
+ Card(int val)
+ Card(const Card& toCopy)
+ ~Card()

+ setVal(int newVal): void
+ getVal(): int
+ getName(): string
+ setName(const string& newName): void

+ getSuit(): string
+ clone(): Card*
+ printCard(): void

+ getFaceDown(): bool
+ setFaceDown(bool newFaceDown): void
+ DisplayCard(): void
+ getColor(): string
```

```
Test
---
+ TestAll(): void
+ TestDeck(): void
+ TestCard(): void
+ TestParticipant(): void
+ TestShufle(): void
+ TestPlay(): void

+ TestMenu(): void
+ TestDisplayCard(): void
+ TestPlayKlondike(): void
```

```
Spade
---
- suit: string
---
+ Spade()
+ Spade(int val)
+ Spade(Spade& toCopy)
+ ~Spade()

+ getSuit(): string
+ clone(): Card*
+ printCard(): void
```

```
Clover
---
- suit: string
---
+ Clover()
+ Clover(int val)
+ Clover(Clover& toCopy)
+ ~Clover()

+ getSuit(): string
+ clone(): Card*
+ printCard(): void
```

```
Heart
---
- suit: string
---
+ Heart()
+ Heart(int val)
+ Heart(Heart& toCopy)
+ ~Heart()

+ getSuit(): string
+ clone(): Card*
+ printCard(): void
```

```
Diamond
---
- suit: string
---
+ Diamond()
+ Diamond(int val)
+ Diamond(Diamond& toCopy)
+ ~Diamond()

+ getSuit(): string
+ clone(): Card*
+ printCard(): void
```

```
Klondike
---
- deck: Deck*
- drawPile: Deck*
- spadeFoundation: Deck*
- heartFoundation: Deck*
- diamondFoundation: Deck*
- cloverFoundation: Deck*
- tableau1: Deck*
- tableau2: Deck*
- tableau3: Deck*
- tableau4: Deck*
- tableau5: Deck*
- tableau6: Deck*
- tableau7: Deck*
---
+ klondike()
+ ~klondike()
+ klondike(klondike& original)
+ operator=(klondike& original): klondike&
+ play(): void
+ startGame(): void
+ showBoard(): void
+ getUserMove(): int
+ drawCard(): void
+ moveCard(): void
+ moveCardCheck(Deck& source, Deck& target, int numCards = 1): bool
+ getTargetDeck(string& initials): Deck*
+ isTableauToTableau(Deck& source, Deck& target): bool
+ win(): bool
```

```
Blackjack
---
- player: Player*
---
- dealer: Dealer*
---
- deck: Deck*
---
- playGame(): int
- startGame(): void
- endGame(int win): void

+ Blackjack()
+ ~Blackjack()
+ play(): void
```

```
Deck
---
- deck: LinkedList*
---
- generator: mt19937
---
+ Deck()
+ ~Deck()
+ Deck(Deck& original)
+ operator=(Deck& original): Deck&

+ fill(): void
+ shuffle(): void
+ pop_back(): Card*
+ size(): int
+ printDeck(): void
+ clearDeck(): void
+ getDeck(): LinkedList*

+ KlondikeFill(): void
+ KlondikePrintDeck(): void
+ MoveLastCardTo(Deck& targetDeck): void
+ MoveCardSequence(Deck& targetDeck, int numCards): void
+ IsValidCardSequence(int startIndex): bool
```

```
Node
---
- data: Card*
---
- next: Node*
- prev: Node*
---
+ Node()
+ Node(Card* data)
+ ~Node()

+ getData(): Card*
+ getNext(): Node*
+ getPrev(): Node*

+ setData(Card* newData): void
+ setNext(Node* newNext): void
+ setPrev(Node* newPrev): void
```

```
LinkedList
---
# head: Node*
# tail: Node*
---
# count: int
---
+ LinkedList()
+ ~LinkedList()

+ push_back(Card* card): void
+ deleteNode(Node* toDelete): void
+ size(): int
+ clear(): void
+ mergeSort(): void
+ getHead(): Node*
+ getTail(): Node*
+ mergeLists(LinkedList* ListB): void
+ setHead(Node* newHead): void
+ setTail(Node* newTail): void
+ setSize(int newSize): void
+ split(LinkedList* left, LinkedList* right): void
+ updateTailAndSize(): void
+ mergeSort(LinkedList* topListPtr): void
+ pop_back(): Card*
```