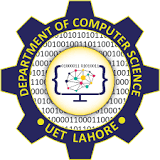
**DSA LAB MID PROJECT**



**Session 2023-2027**

**Submitted by:**

Umme Aymen 2023-CS-112

**Supervised by:**

Sir Nazeef ul Haq

**Course:**

Data Structure and Algorithm

**Department of Computer Science**

**University of Engineering and Technology,**

**Lahore Pakistan**

Table of Contents

[**1 Project Description** 3](#_Toc182161958)

[**2. Key Features** 3](#_Toc182161959)

[**2.1 Game Mechanics:** 3](#_Toc182161960)

[**2.2 Validation and Testing:** 3](#_Toc182161961)

[**2.3 Scoring, Timer and Undo/Redo Functionality:** 3](#_Toc182161962)

[**3 Data Structures** 3](#_Toc182161963)

[**3.1 Stack(LIFO):** 3](#_Toc182161964)

[**3.2 Queue(FIFO):** 3](#_Toc182161965)

[**3.3 Array:** 3](#_Toc182161966)

[**3.4 Linked list:** 3](#_Toc182161967)

[**3.5 HashMap:** 4](#_Toc182161968)

[**4 Technical Details** 4](#_Toc182161969)

[**4.1 Classes:** 4](#_Toc182161970)

[**4.2 UI Components:** 4](#_Toc182161971)

[**5 Game Rules** 4](#_Toc182161972)

[**5.1 Objective** 4](#_Toc182161973)

[**5.2 Setup** 4](#_Toc182161974)

[**5.3 Game Play** 5](#_Toc182161975)

[**6. GitLab Repository:** 6](#_Toc182161976)

[**7. Wire Frames:** 6](#_Toc182161977)

# **1 Project Description**

This project is about solitaire game build on Klondike rules. This project is built using Tkinter framework for frontend. Backend is implemented using Data Structures such as linked list, array, queue, stack and others for the proper functioning and implementation of the game. It includes features like card shuffling, card movement, valid moves and win condition check.

# **2. Key Features**

## **2.1 Game Mechanics:**

In **Game Mechanics** for Solitaire, gameplay begins by shuffling the deck and dealing seven tableau piles, each with one more card than the last, and only the last card face-up. **Moves** allow cards to transfer between tableau piles, the stockpile, and foundation piles based on color and rank rules. The **Foundation** requires players to build piles from Ace to King for each suit. The **Victory Condition** is achieved when all cards are successfully moved to the foundation, completing each suit in ascending order.

## **2.2 Validation and Testing:**

The game ensures **legal moves** by validating that cards are moved according to the rules, such as alternating colors and sequential ranks. **Card flipping** is tested to ensure that cards are flipped correctly after being moved from tableau piles. The **win condition** is verified to check that the game accurately identifies when all cards are successfully moved to the foundation piles. **Game flow** is thoroughly tested from initialization to completion, ensuring that all rules are followed and the game progresses correctly through each phase.

## **2.3 Scoring, Timer and Undo/Redo Functionality:**

The **Undo/Redo system** uses stacks to track moves, enabling players to reverse actions. A **timer and scoring system** tracks gameplay duration and scores based on valid moves, enhancing the gaming experience.

# **3 Data Structures**

## **3.1 Stack(LIFO):**

Stack is used to model piles in Tableau and Foundation piles. It is used to implement face up the top card. It has been used in undo redo functionality.

## **3.2 Queue(FIFO):**

Queue is used to initialize the deck where the first card is removed from deck and it is inserted at the last if it is not moved from deck to tableau or foundation pile.

## **3.3 Array:**

In deck, the cards are implemented in array and stack cards are also included in array.

## **3.4 Linked list:**

Linked list is used in stack for cards implementation and tableau piles also include a linked list for storing all the new cards inserted in the pile.

## **3.5 HashMap:**

Built in Hash maps is used to implement the ranks as key values they are used in movement validations.

# **4 Technical Details**

## **4.1 Classes:**

Following classes and attributes

|  |  |  |
| --- | --- | --- |
| Classes | Attributes | Behaviours |
| 1. Card | Suit, rank, faceup, color | FlipCard(), load\_image() |
| 1. Deck | cards[],suits[],ranks[] | InitializeDeck(), draw\_card(), is\_empty(), shuffle() |
| 1. Foundation | FoundationPiles[stacks()\*4] | Add\_card(), get\_top\_card(), remove\_card() |
| 1. Tableau | TableuPiles[\*7], deck | Initialize(), add\_card(), get\_topcard(), get\_pile(), remove\_pile(), is\_column \_empty() |
| 1. Circular Queue | Deck, top, rear, card[] | Initialize(), enqueue(), dequeue(), peek() |
| 1. Linked List | Head,size | Insert(),retrieveWholelist(),retrieveFromSpecificIndex() |
| 1. Stack | Top, items[] | Push(),pop(),getCards(),peek() |

## **4.2 UI Components:**

Following UI components are used

1. Card
2. Deck
3. Foundations
4. Labels
5. Tableau
6. Events

# **5 Game Rules**

Here are the standard rules for playing the classic Solitaire Klondike game:

## **5.1 Objective**

The goal of Solitaire Klondike is to build four foundation piles, each with cards of the same suit, in ascending order from Ace to King.

## **5.2 Setup**

1. Tableau: The game starts with seven piles of cards laid out in a cascading format:

The first pile has one card, the second has two, the third has three, and so on until the seventh pile has seven cards.

Only the top card in each pile is turned face-up, while the others remain face-down.

2. Foundations: There are four foundation piles, one for each suit (hearts, diamonds, clubs, and spades). These start empty and are built up in ascending order, beginning with the Ace.

3. Stockpile: The remaining cards form the stockpile, which is used to draw additional cards as needed.

## **5.3 Game Play**

**1. Moving Cards in the Tableau:**

Cards in the tableau can be moved onto another card if they are one rank lower and of the opposite color (e.g., a red 5 can be placed on a black 6).

You can also move groups of cards, provided they are arranged in descending order and alternate colors.

Empty tableau piles can be filled only with a King or a sequence of cards starting with a King.

**2. Building Foundations:**

Only an Ace can be placed in an empty foundation pile.

You build each foundation pile by placing cards of the same suit in ascending order (Ace, 2, 3, …, King).

**3. Using the Stockpile:**

Draw cards from the stockpile to add to your playable cards. The way you draw depends on the specific variation:

Draw 1: Draw one card at a time, cycling through the deck as needed.

Draw 3: Draw three cards at a time, only using the top card in each draw pile. If you reach the end, you can go through the stockpile again.

**4. Winning the Game:**

The game is won when all cards are successfully moved to the foundation piles in the correct order and suit.

**Additional Rules:**

You can cycle through the stockpile as many times as allowed by the variation you’re playing (some limit the cycles to increase difficulty).

Strategic planning is essential. Sometimes, it’s beneficial to keep cards in the tableau rather than immediately placing them in the foundations.

# **6. GitLab Repository:**

You can access the GitLab repository from following link:

https://gitlab.com/ummeaymen499/CSC200M24PID112

# **7. Wire Frames:**

