README – Pokémon Card Collection Tracker

1. Overview

The Pokémon Card Collection Tracker is a C++ console program that helps users manage a small personal inventory of Pokémon trading cards.

The program allows users to:

- **Add new Pokémon cards**
- **Remove cards**
- **Search cards by name**
- **Sort the collection by card value (using Bubble Sort)**
- **Display all stored cards**

This project demonstrates structured data management, searching, sorting, and user-driven menu operations.

---

2. Design Decisions

Structured Data (Structs)

I used a struct named PokemonCard to group:

- **card name**
- **category**
- **value**
- **condition**

Using a struct makes the data organized and easier to pass to functions.

Programming Constructs Used

The project uses:

- **arrays**
- **loops**
- **conditionals**
- **functions**
- **structs**
- **input validation**

These are required components of the assignment and help keep the code modular and readable.

Searching Method

- **Linear Search**
  I used linear search because:
    - **the list is small**
    - **it is simple to implement**
    - **sorting is not required before searching**
      It scans each element until the card is found.

Sorting Method

- **Bubble Sort (ONLY)**
  Bubble sort was chosen because:
- **the assignment requires a sorting algorithm**
- **bubble sort is easy to implement**

- **the dataset is small, so performance is not an issue**

The sort organizes the cards by value from lowest to highest.

Data Persistence

No file usage was required, so data is stored only in memory during program execution. When the program closes, the data resets.

Alternative Approaches Considered

I considered:

- **selection sort**
- **binary search**
- **file storage**

I did not use them because:

- **bubble sort was sufficient**
- **binary search requires sorted data, while linear search works without sorting**
- **assignment instructions allowed programs without file I/O**

---

3. Testing Summary

Testing Log

Test Case	Input	Expected Output	Actual Output	Pass/Fail
Add Card	Pikachu, Electric, 50, Mint	Card added successfully	Same	Pass

Search Existing Card	**Pikachu**	**Found at index X**	**Same**	**Pass**
Search Missing Card	**Charmander**	**"Card not found"**	**Same**	**Pass**
Remove Existing Card	**Pikachu**	**Pikachu removed**	**Same**	**Pass**
Remove Non-existent Card	**MissingName**	**Error message**	**Same**	**Pass**
Sort Cards	**Values: 40, 10, 30**	**Sorted: 10, 30, 40**	**Same**	**Pass**
Bad Input (string for value)	**"abc"**	**Ask to re-enter**	**Same**	**Pass**

Testing Methods Used

- **Manual testing for each menu option**
- **Edge-case testing (empty list, full list, incorrect inputs)**
- **Repeated add/remove cycles to check stability**

Example Invalid Inputs

- **Entering letters for value → program re-prompts**
- **Removing card when list is empty → prints error**
- **Adding card when list is full → prints warning**

---

4. Technical Walkthrough

Main Functionalities

- **addCard() – adds a new Pokémon card to the array**

- **removeCard() – removes card by searching its name**
- **displayCards() – prints all stored cards**
- **linearSearch() – finds card index by name**
- **bubbleSort() – sorts by value (ascending)**
- **displayMenu() – shows user options**

Program Flow

1. **Program displays menu**
2. **User selects an action**
3. **Program executes function**
4. **Menu repeats until user quits**

Video Demonstration

https://youtu.be/ouZq2vbKnck

---

5. Challenges and Lessons Learned

Challenges

- **Designing the menu loop and preventing input crashes**
- **Deciding between sorting algorithms**
- **Managing array size limits**
- **Removing elements by shifting the array**

Lessons Learned

- **Structs make data cleaner and easier to manage**
- **Modularity (functions) prevents messy and repetitive code**
- **Sorting and searching algorithms are useful in real programs**
- **Planning the program flow saves time in debugging**

---

6. Future Improvements

If I had more time, I would add:

- **File storage to save data between runs**
- **Binary search (after sorting) for faster lookups**
- **Ability to edit card details**