**Technical Design Document**

**Name:** Matthew Pocrnic  
 **Date Created:** 5/18/2025

### **Program Description:**

This program simulates a cinema ticket sale where only 20 tickets are available in total. Each customer can purchase a maximum of 4 tickets. The program continues to prompt users to enter the number of tickets they wish to purchase until all tickets are sold. Then it displays the total number of buyers who successfully purchased tickets.

### **Functions used in the Program (list in order as they are called):**

**1. Function Name:** getTickets  
 **Description:** Validates the number of tickets a user wants to purchase based on several conditions including minimum, maximum per person, and remaining tickets.  
 **Parameters:**

* wanted (int) – The number of tickets the user wants to purchase
* remaining (int) – The number of tickets still available for sale  
   **Variables:** No internal variables besides parameters

**Logical Steps:**

1. Check if wanted is less than 1; if so, display an error and return 0
2. Check if wanted exceeds MAX\_PER\_PERSON; if so, display an error and return 0
3. Check if wanted exceeds remaining; if so, display an error and return 0
4. If all conditions pass, return the value of wanted

**Returns:**Returns the number of tickets to be purchased if the request is valid, otherwise returns 0

**2. Function Name:** main  
**Description:** Manages the ticket selling loop, tracks the number of tickets sold and total buyers, and handles user input.  
**Parameters:** None

**Variables:**

* ticketsLeft (int) – Tracks the remaining number of tickets
* totalBuyers (int) – Counts how many users successfully bought tickets
* wantedTickets (int) – Stores the user's input for number of tickets they want
* bought (int) – Stores the result from the getTickets function

**Logical Steps:**

1. Initialize ticketsLeft to MAX\_TICKETS and totalBuyers to 0
2. Display a welcome message
3. Use a loop that continues while ticketsLeft > 0
4. Prompt user to enter number of tickets
5. Validate input using try-except to catch invalid integers
6. Call getTickets() to check if the request is valid
7. If valid, update ticketsLeft and increment totalBuyers
8. When all tickets are sold, print total number of buyers

**Returns:** None

### **Logical Steps (Program Flow):**

1. Start the program by calling main()
2. main() initializes the ticket count and buyer count
3. Inside a loop, ask the user how many tickets they want
4. Call getTickets() to validate the request
5. If valid, update counts and display confirmation
6. Continue until all tickets are sold
7. Display total number of buyers and end program

**Link to my COP2373 repository:** [**here**](https://github.com/mpocrnic/COP2373)

Screenshot of output from running code

