

Project - 1 MA517M-Basic Programming Laboratory Last Date: 09 November 2025
Name _____ Roll No.: MA25M102

Cantor Set Construction using C++ Classes and Operator Overloading

Objective: To design a C++ program that constructs and manipulates the *Cantor set* using recursion, object-oriented programming principles, and operator overloading.

The Cantor set is constructed by repeatedly removing the open middle third of each interval, starting with the closed interval $[0, 1]$. After k steps, the set consists of 2^k disjoint intervals, each of length 3^{-k} .

Problem Description

Design a class `CantorSet` that models the k -th stage of the Cantor set construction. The class should store the collection of intervals that remain after k iterations, and provide overloaded operators for set operations and comparisons.

Mathematical Background

- Start with $C_0 = [0, 1]$.
- Define recursively:

$$C_{k+1} = \frac{1}{3}C_k \cup \left(\frac{2}{3} + \frac{1}{3}C_k\right)$$

- The limit set $C = \bigcap_{k=0}^{\infty} C_k$ is the **Cantor set**.

Class Specification

- **Class Name:** `CantorSet`
- **Private Data Members:**

```
– int level; // level of iteration
– std::vector<std::pair<double,double>> intervals; // remaining intervals
```

- **Public Member Functions:**

```
– CantorSet(int k=0); // constructor builds the k-th stage
– void generate(); // recursively generates the intervals
– void display() const; // prints intervals in [a, b] form
– int getLevel() const; // returns the iteration level
```

Operator Overloading

- operator `+`() Union of two Cantor sets:

$$C_k + C_m = C_{\max(k,m)} \cup C_{\min(k,m)}$$

- operator `==`() Checks whether two Cantor sets are at the same level (and have identical intervals).
- operator `<`() and operator `>`() Compare Cantor sets by level of iteration (i.e., their depth of construction).

Tasks

1. Create the Cantor sets C_1 , C_2 , and C_3 .
2. Display the intervals for each iteration.
3. Combine C_2 and C_3 using operator `+` and display the resulting intervals.
4. Compare C_1 , C_2 , and C_3 using the overloaded `<`, `>`, and `==` operators.

Expected Output Example

C1 Intervals:

[0, 0.3333] [0.6667, 1]

C2 Intervals:

[0, 0.1111] [0.2222, 0.3333] [0.6667, 0.7778] [0.8889, 1]

C3 Intervals:

[0, 0.037037] [0.074074, 0.111111] ... [0.962963, 1]

Union (C2 + C3) constructed.

C1 < C2 : True

C3 == C3 : True

Project - 2: Number Slide Game Using C++ Classes

Problem Statement

Design and implement a **Number Slide Game** using C++ classes. The program should simulate a 4×4 sliding puzzle where the goal is to arrange numbers from 1 to 15 in order, leaving one space. The project should utilize object-oriented programming concepts such as classes, objects, encapsulation, and methods for handling game logic.

Project Requirements

1. Create a **Matrix** class to represent the 4×4 game board.
 - (a) Include a method to generate a random matrix of size 4×4 using `srand()` in the range [1, 16].
 - (b) Include a method to check that all entries of the matrix are distinct.
 - (c) Remove the entry with the value 16 to represent the space.
2. Create a **Game** class to manage gameplay.
 - (a) Display options for the user: **Play** or **Solution**.
 - (b) If the user chooses **Play**, show the navigation commands:
 - A/a for Left
 - S/s for Down
 - D/d for Right
 - W/w for Up
 - Q/q for Quit
 - (c) When the user presses Q/q, confirm again before quitting.
 - (d) After each move, display the updated matrix in a nicely formatted way.
3. If the user chooses **Solution**, display the sequence of movements leading to the solved puzzle, and show the final output.
4. Ensure proper encapsulation of game logic and board operations within the respective classes.

Suggested Class Structure

1. **Matrix** Class:
 - Data member: 4×4 integer array representing the board
 - Methods: `generateMatrix()`, `checkDistinct()`, `removeEmpty()`, `displayMatrix()`
2. **Game** Class:
 - Data member: **Matrix** object, user choice
 - Methods: `playGame()`, `showSolution()`, `processMove(char move)`, `confirmQuit()`

Reference

For more details about the game, visit: <http://www.artbylogic.com/puzzles/numSlider/numberShuffle.htm>