# Fractol - 42 Project

## Overview

Fractol is a graphical project that renders fractals using MiniLibX. The program supports the Mandelbrot set, Julia set, and an additional bonus fractal. Users can interact with the fractals by zooming, changing colors, and modifying iterations.

## Features

* Supports Mandelbrot, Julia, and a bonus fractal.
* Mouse zooming and color shifting.
* Arrow keys for navigation.
* Custom Julia constants can be provided as arguments.

## Compilation and Execution

To compile the project, run:

make

To execute Fractol with a fractal:

./fractol <fractal\_name> [cx cy]

Example usages:

./fractol mandelbrot

./fractol julia -0.7 0.27015

./fractol fractal\_bonus

## File Structure

fractol/

│── srcs/

│ ├── main.c # Entry point

│ ├── mandelbrot.c # Mandelbrot fractal calculations

│ ├── julia.c # Julia set calculations

│ ├── start\_julia.c # Initialization for Julia

│ ├── start\_mandelbrot.c # Initialization for Mandelbrot

│ ├── start\_fractal\_bonus.c # Initialization for bonus fractal

│ ├── start\_fractals.c # General fractal window setup

│ ├── mouse\_and\_keys.c # Key and mouse controls

│ ├── mouse\_and\_keys\_bonus.c # Advanced zooming & interactions

│ ├── event.c # Event handling for keys and mouse

│ ├── fractal\_bonus.c # Bonus fractal implementation (Tricorn)

│ ├── fractol.c # Core fractal rendering logic

│ ├── utils.c # Utility functions

│── includes/

│ ├── fractol.h # Header file with function prototypes

│── Makefile # Compilation rules

## Controls

| Key | Action |
| --- | --- |
| Arrow Keys | Move the view |
| Scroll | Zoom in/out |
| C | Change colors |
| R | Reset view |
| ESC | Close the program |

## Functions

### main.c

* print\_usage() - Displays the usage instructions.
* main(argc, argv) - Parses arguments and starts the correct fractal.

### mandelbrot.c

* init\_mandelbrot\_vars() - Initializes Mandelbrot calculations.
* mandelbrot\_iterations() - Computes the number of iterations for Mandelbrot.
* draw\_mandelbrot\_pixel() - Colors a pixel based on iterations.
* draw\_mandelbrot() - Renders the Mandelbrot fractal.

### julia.c

* init\_julia\_vars() - Initializes Julia fractal calculations.
* julia\_iterations() - Computes the Julia iterations.
* draw\_julia\_pixel() - Colors a Julia pixel.
* draw\_julia() - Renders the Julia fractal.

### fractal\_bonus.c

* init\_vars() - Initializes fractal variables.
* calculate\_fractal() - Performs calculations for bonus fractal (Tricorn).
* draw\_fractal\_bonus() - Draws the bonus fractal.

### event.c

* key\_hook() - Handles keyboard inputs.
* mouse\_hook() - Handles mouse zooming.

### mouse\_and\_keys.c

* change\_iterations() - Modifies iteration count.
* init\_fractal() - Initializes fractal settings.
* generate\_random\_c() - Generates random values for Julia.
* handle\_mouse() - Handles zooming.
* set\_random\_julia() - Sets Julia constants randomly.

### fractol.c

* get\_color() - Generates color based on iteration count.
* draw\_fractal() - Calls the correct fractal drawing function.

### utils.c

* close\_window() - Cleans up and exits the program.

### start\_fractals.c

* init\_and\_create\_window() - Initializes MiniLibX and creates a window.
* init\_image\_data() - Retrieves image data buffer.
* start\_fractal() - Initializes and starts a fractal window.

## Bonus Features

* Additional fractal support (fractal\_bonus).
* Zooming follows the mouse.
* Color shifting.
* Improved navigation and rendering optimizations.