Christopher Back’s fitness tracker - WIP

Thought process

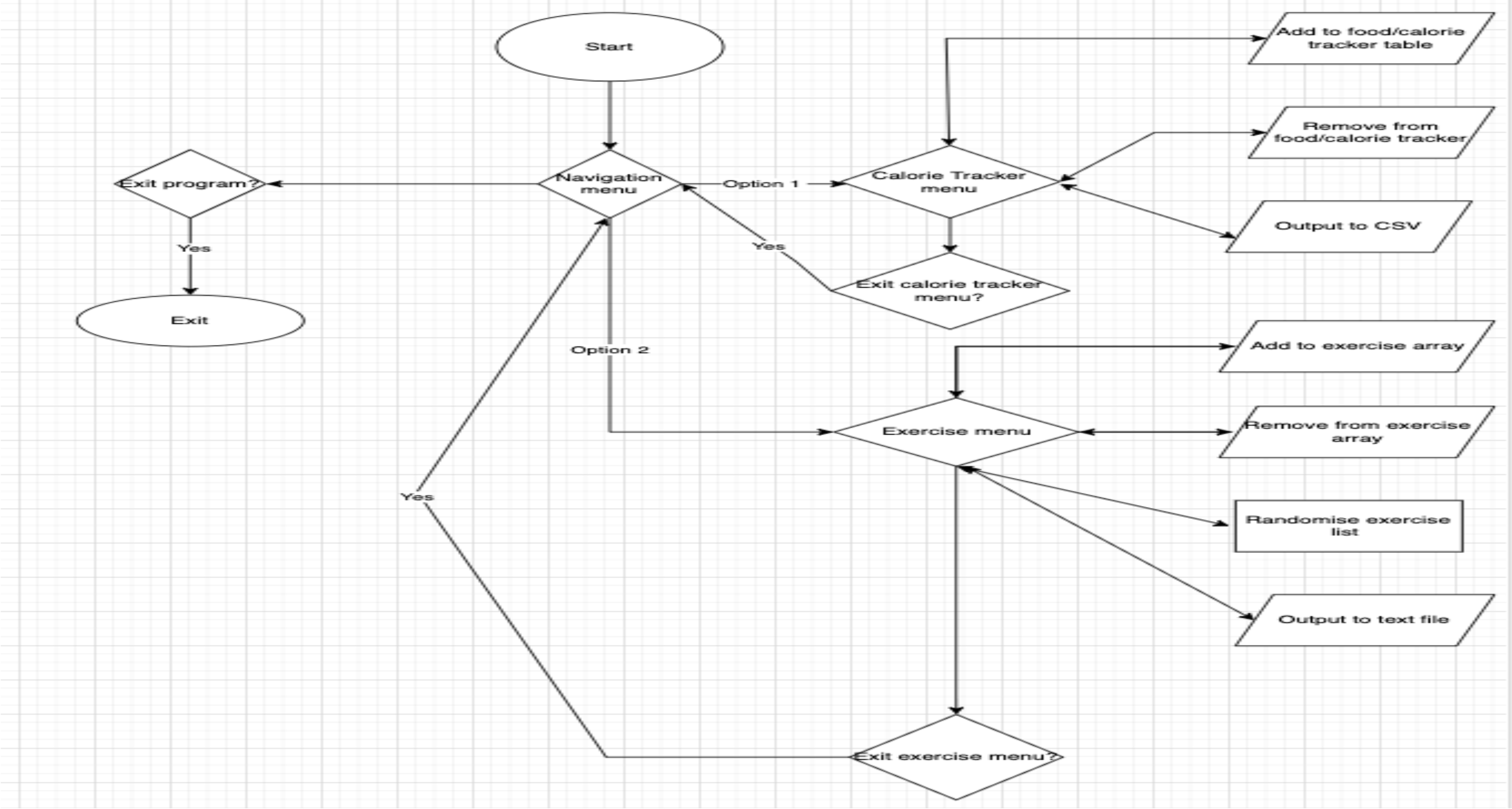
Wanted to create an app that helped track fitness goals due to myself gaining weight during COVID.

To lose weight, best thing to do is track food intake and calories and also exercise.

Created a program that can do both, and for exercise since you sometimes get bored doing same program, planning to create a randomizer.

Randomizer is good because if you do same program for more than 6-8 weeks, it loses its effectiveness.

User flow diagram



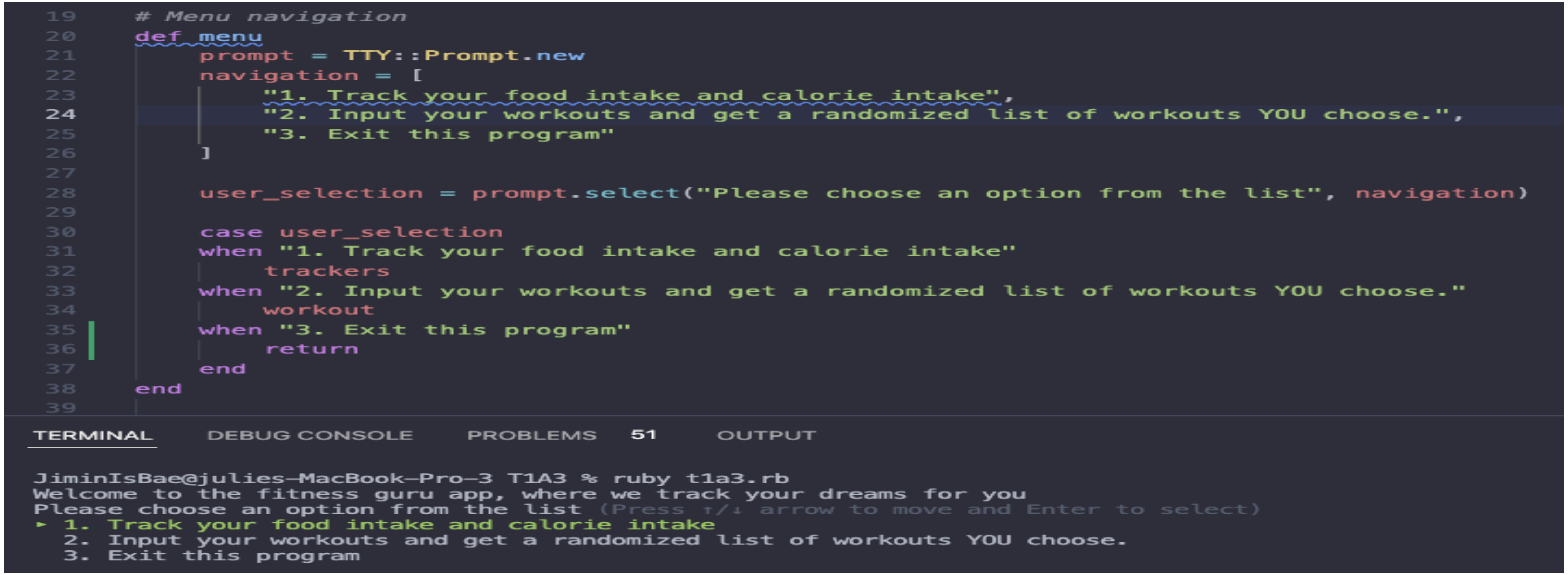
Gems used

- Tty-prompt

- Tty-table

- Rainbow

Navigation menu

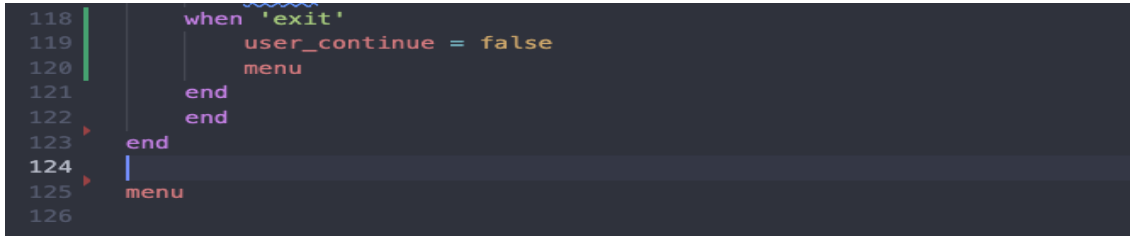


```ruby
# Menu navigation
def menu
    prompt = TTY::Prompt.new
    navigation = [
        "1. Track your food intake and calorie intake",
        "2. Input your workouts and get a randomized list of workouts YOU choose.",
        "3. Exit this program"
    ]

    user_selection = prompt.select("Please choose an option from the list", navigation)

    case user_selection
    when "1. Track your food intake and calorie intake"
        trackers
    when "2. Input your workouts and get a randomized list of workouts YOU choose."
        workout
    when "3. Exit this program"
        return
    end
end
```

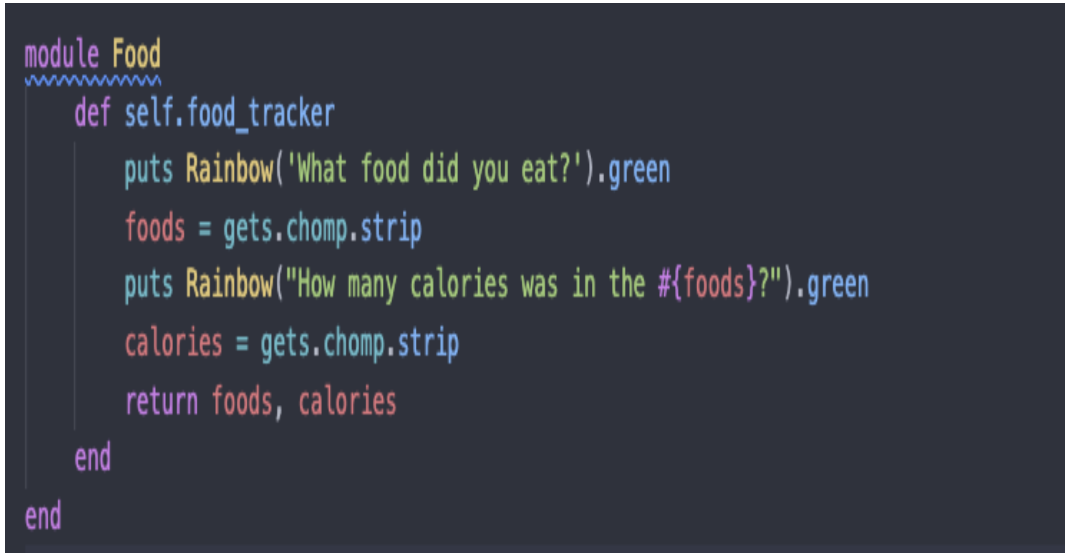
```
JiminIsBae@julies-MacBook-Pro-3 T1A3 % ruby t1a3.rb
Welcome to the fitness guru app, where we track your dreams for you
Please choose an option from the list (Press ↑/↓ arrow to move and Enter to select)
‣ 1. Track your food intake and calorie intake
  2. Input your workouts and get a randomized list of workouts YOU choose.
  3. Exit this program
```



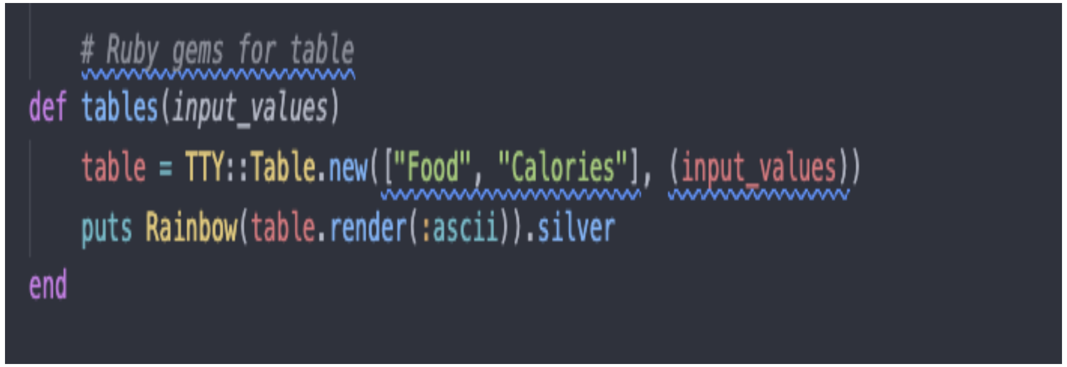
```ruby
    when 'exit'
        user_continue = false
        menu
    end
    end
end

menu
```

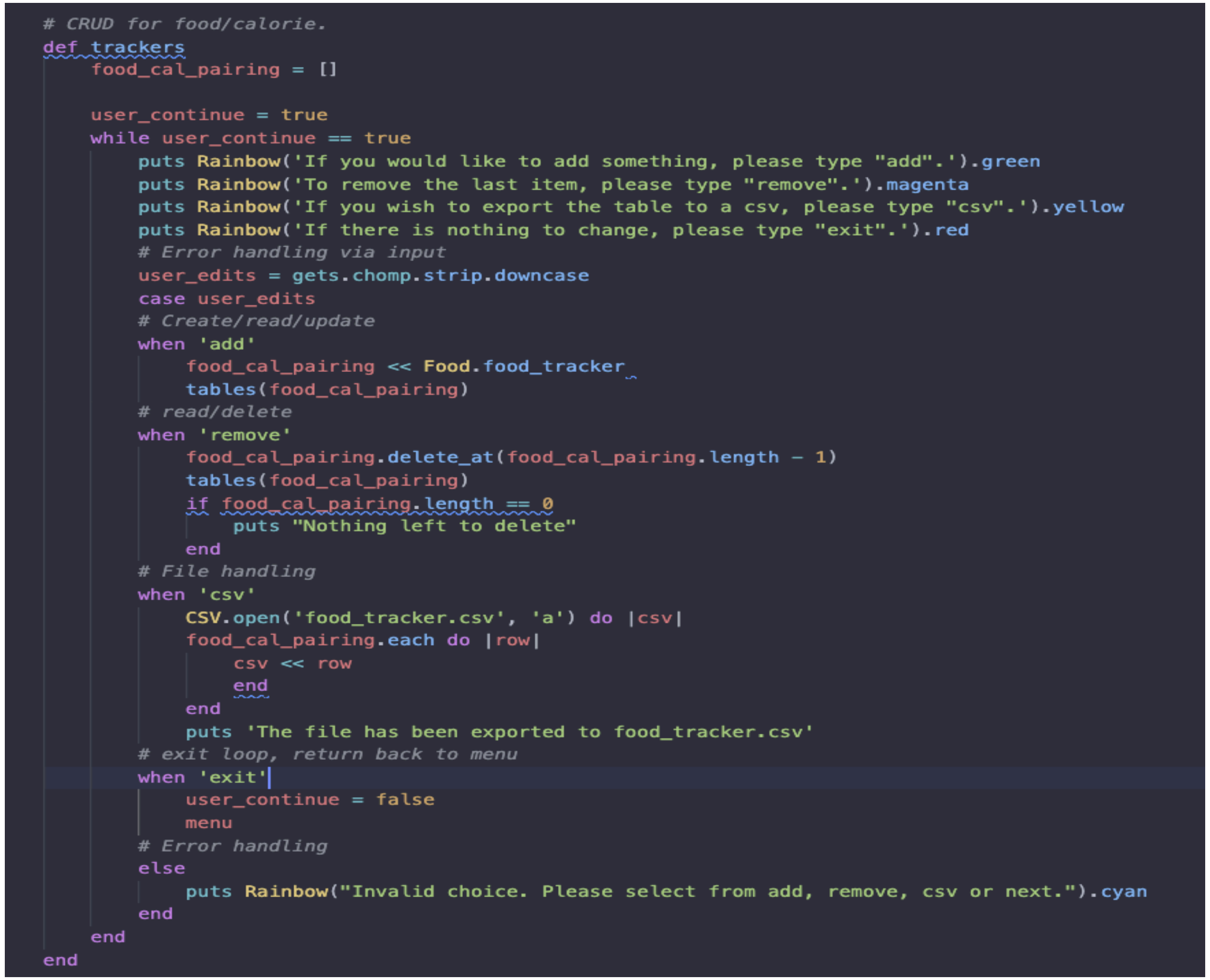
Food and calorie tracker (WIP)



```ruby
module Food
  def self.food_tracker
    puts Rainbow('What food did you eat?').green
    foods = gets.chomp.strip
    puts Rainbow("How many calories was in the #{foods}?").green
    calories = gets.chomp.strip
    return foods, calories
  end
end
```



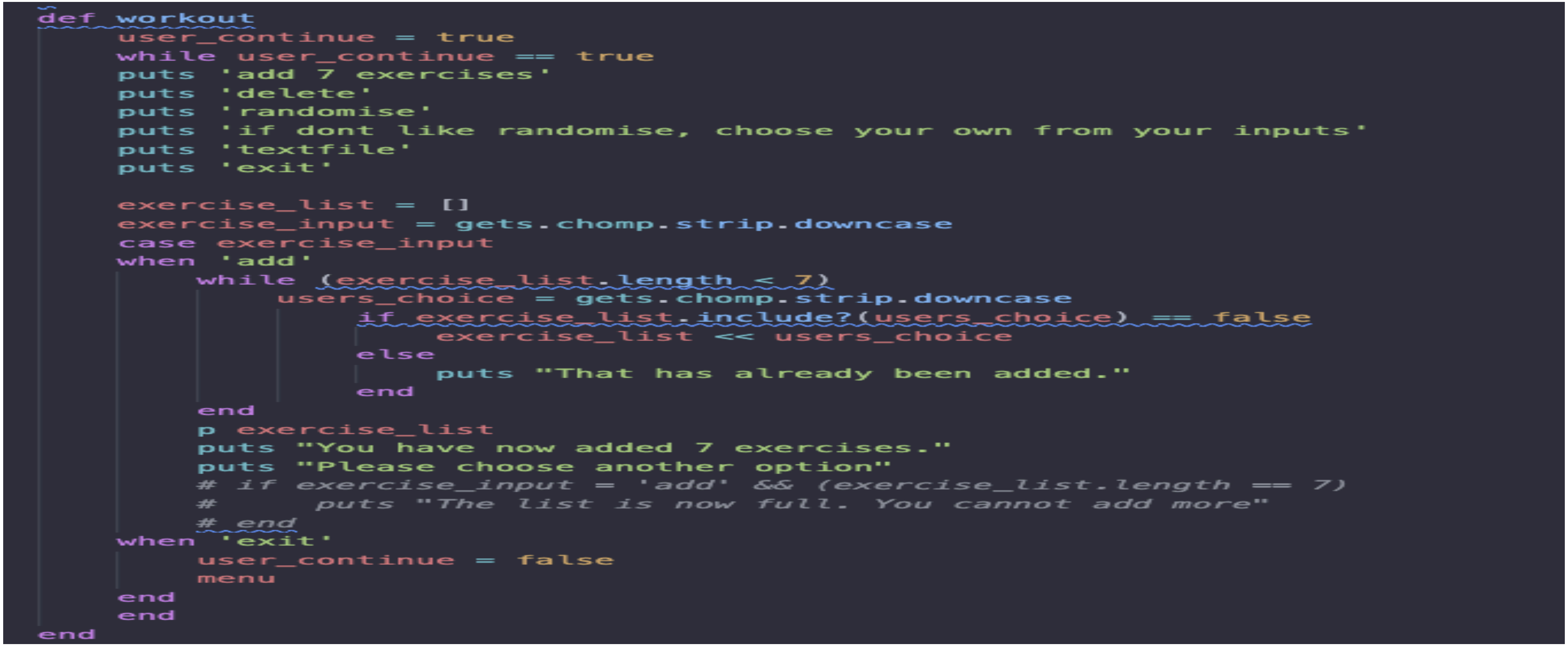
```ruby
  # Ruby gems for table
def tables(input_values)
  table = TTY::Table.new(["Food", "Calories"], (input_values))
  puts Rainbow(table.render(:ascii)).silver
end
```



```ruby
# CRUD for food/calorie.
def trackers
  food_cal_pairing = []

  user_continue = true
  while user_continue == true
    puts Rainbow('If you would like to add something, please type "add".').green
    puts Rainbow('To remove the last item, please type "remove".').magenta
    puts Rainbow('If you wish to export the table to a csv, please type "csv".').yellow
    puts Rainbow('If there is nothing to change, please type "exit".').red
    # Error handling via input
    user_edits = gets.chomp.strip.downcase
    case user_edits
    # Create/read/update
    when 'add'
      food_cal_pairing << Food.food_tracker
      tables(food_cal_pairing)
    # read/delete
    when 'remove'
      food_cal_pairing.delete_at(food_cal_pairing.length - 1)
      tables(food_cal_pairing)
      if food_cal_pairing.length == 0
        puts "Nothing left to delete"
      end
    # File handling
    when 'csv'
      CSV.open('food_tracker.csv', 'a') do |csv|
      food_cal_pairing.each do |row|
        csv << row
        end
      end
      puts 'The file has been exported to food_tracker.csv'
    # exit loop, return back to menu
    when 'exit'
      user_continue = false
      menu
    # Error handling
    else
      puts Rainbow("Invalid choice. Please select from add, remove, csv or next.").cyan
    end
  end
end
```

Workout (WIP)



```ruby
def workout
    user_continue = true
    while user_continue == true
    puts 'add 7 exercises'
    puts 'delete'
    puts 'randomise'
    puts 'if dont like randomise, choose your own from your inputs'
    puts 'textfile'
    puts 'exit'

    exercise_list = []
    exercise_input = gets.chomp.strip.downcase
    case exercise_input
    when 'add'
        while (exercise_list.length < 7)
            users_choice = gets.chomp.strip.downcase
                if exercise_list.include?(users_choice) == false
                    exercise_list << users_choice
                else
                    puts "That has already been added."
                end
        end
        p exercise_list
        puts "You have now added 7 exercises."
        puts "Please choose another option"
        # if exercise_input = 'add' && (exercise_list.length == 7)
        #     puts "The list is now full. You cannot add more"
        # end
    when 'exit'
        user_continue = false
        menu
    end
    end
end
```