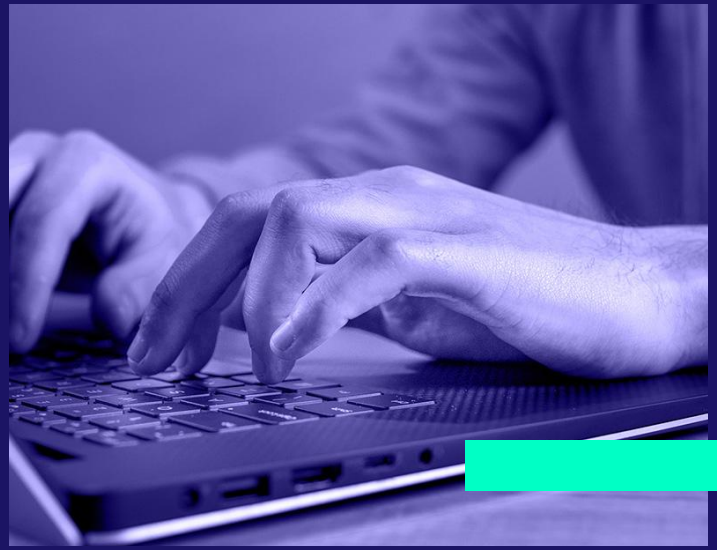
Terminal App

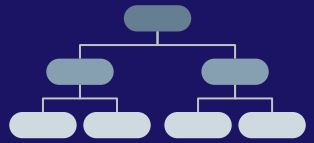
By Krish

Roulette App

A simple Ruby Terminal Application that acts as a form of entertainment. Similar to Roulette at a casino. Player tries to guess a number and gets rewarded with credits if they are correct.

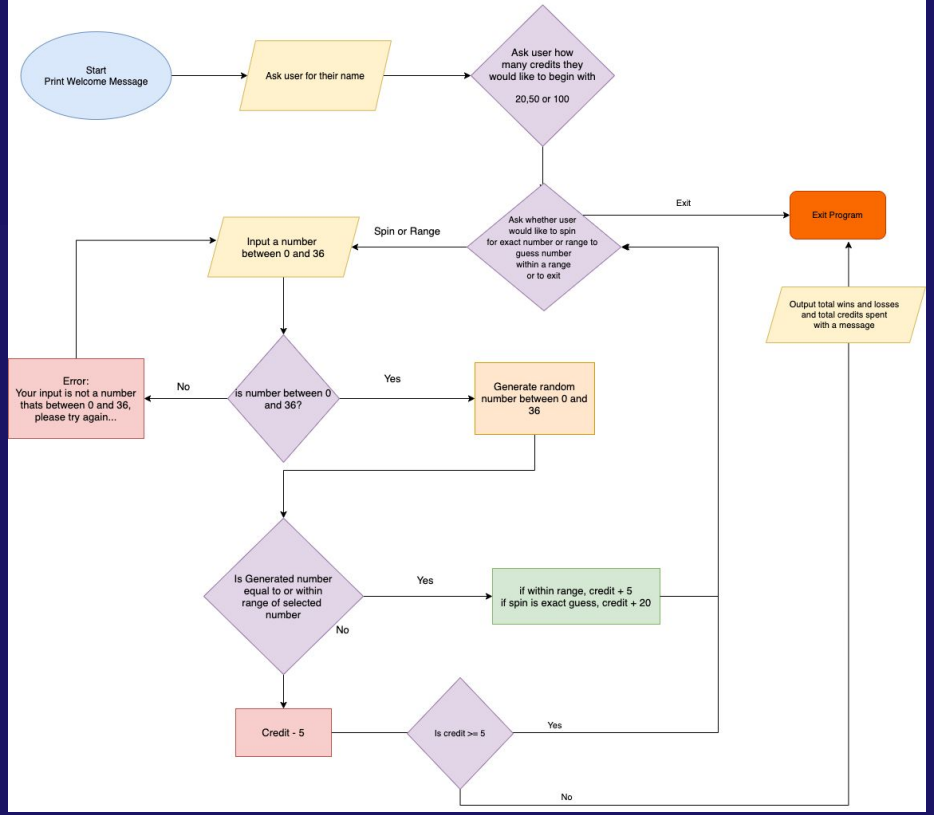


FlowChart



Roulette App

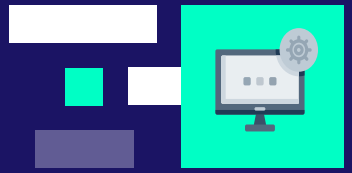
A simple app that allows a user to select how many credits they would like to play, and then gives them an option on whether they would like to guess the exact number or guess within a range of numbers. If the user gets the range correct, they receive 5 credits, if they get the spin correct, they get 20 credits, if they guess incorrectly for either, they lose 5 credits. The user can exit at anytime, or the game will stop once the user is out of credit



App Overview

- Allows a user to select how many credits they would like to play with.
- Options are to guess the exact number (spin) or guess within a range of numbers (range).
- Number within the range gets 5 credits, getting the exact number gets 20 credits, incorrect guess for either option, they lose 5 credits.
- The user can exit at anytime, or the game will stop once the user is out of credit.

Features

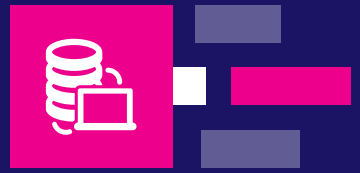


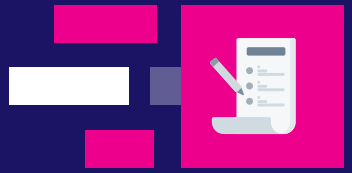
Random Number Generator

generates a random number between 0 and 36

Number Range Validator

validates whether number entered is within a specific range



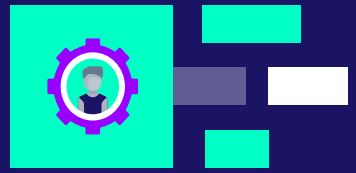


Number History Viewer

Lists of past generated numbers to guide user in deciding what number to pick next

Total Spent

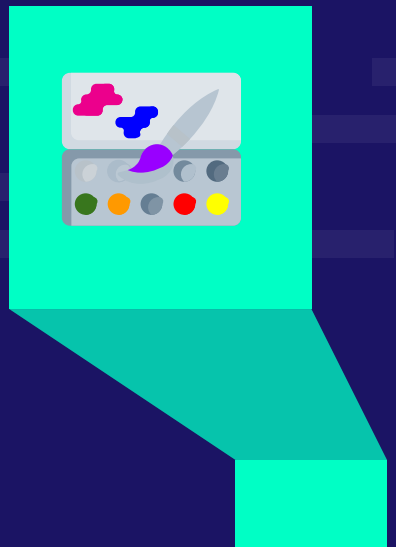
A feature that calculates the total credits spent and displays a corresponding message.



Gems

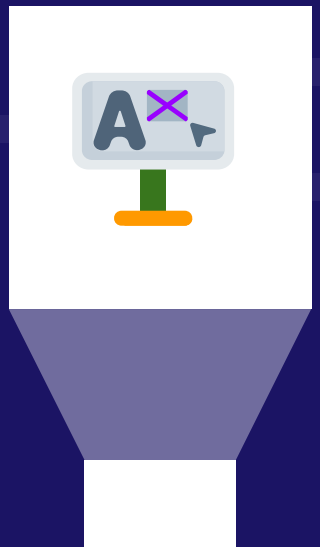
Colorize

Add meaningful colours to code to make for a better user experience, ie turn code red for error messages.



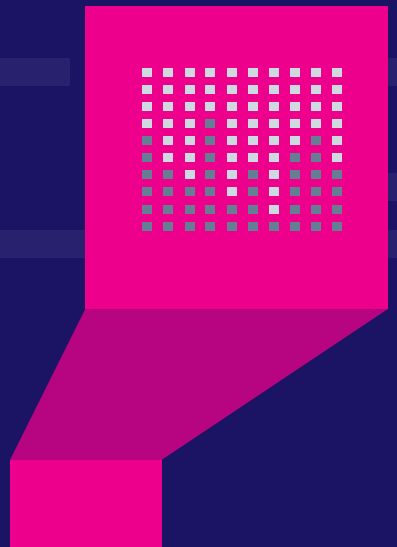
Tty-prompt

Interactive menu to allow for a better user experience allowing the user to select from a menu rather than needing to type out an input each time.



artii

Ascii art for welcome message upon application startup to make for a pleasant entrance



Code Overview

methods.rb — roulette-app

EXPLORER

ROULETTE-APP
- docs
- ppt
- src
  - main.rb M
  - methods.rb M
  - test.rb
- Gemfile
- Gemfile.lock
- README.md

main.rb M | Gemfile.lock | Gemfile | methods.rb M

src > main.rb

You, 38 minutes ago | 1 author (You)

```ruby
require_relative './methods.rb'
require('colorize')
require('tty-prompt')
require('artii')

prompt = TTY::Prompt.new
a = Artii::Base.new

num_list = []

#Asks user to input their name, and stores into name variable
puts a.asciify("Welcome")
puts a.asciify("to")
puts a.asciify("Roulette")
begin
puts "What is your name?".colorize(:blue)
name = gets.chomp
validate_name(name)
puts "Hello there, #{name}!".colorize(:yellow)
rescue InvalidNameError
    puts "Please enter a valid name"
    retry
end


#Asks user how many credits they would like to play with
credit_select = prompt.select("How many credits would you like to begin with?", %w(20 50 100)).to_
credit = credit_select.to_i

#Run credits method
```

methods.rb M

src > methods.rb

You, 6 hours ago | 1 author (You)

```ruby


def credits(credit)

# credit = amount
    case
        when credit < 5
            puts "5 credits is the minimum"
        when credit % 5 != 0
            puts "Please enter a credit that is a multiple of 5 (ie. 5,10,15 etc..)"
            credit = 0
        else
            puts "You have entered #{credit} credits"
    end
end


def num_validator(selected_num)
    if selected_num < 0 || selected_num > 36
        raise ArgumentError

    elsif selected_num.between?(0,36) == false
        raise ArgumentError

    else
        puts "You have selected #{selected_num}".colorize(:blue)
    end
end
```

You, 5 days ago • Added loop so user can keep spinning the wheel un…

PROBLEMS | OUTPUT | TERMINAL | DEBUG CONSOLE

```
krishal@Krishals-MacBook-Pro roulette-app % ruby ./src/main.rb
 _____ _____ _____ _____ _____ _____ _____ _____ _____
|W|e|l|c|o|m|e|
 _____ _____ _____
|t|o|
 _____ _____ _____ _____ _____ _____ _____ _____ _____ _____
|R|o|u|l|e|t|t|e|
What is your name?
```

OUTLINE

TIMELINE

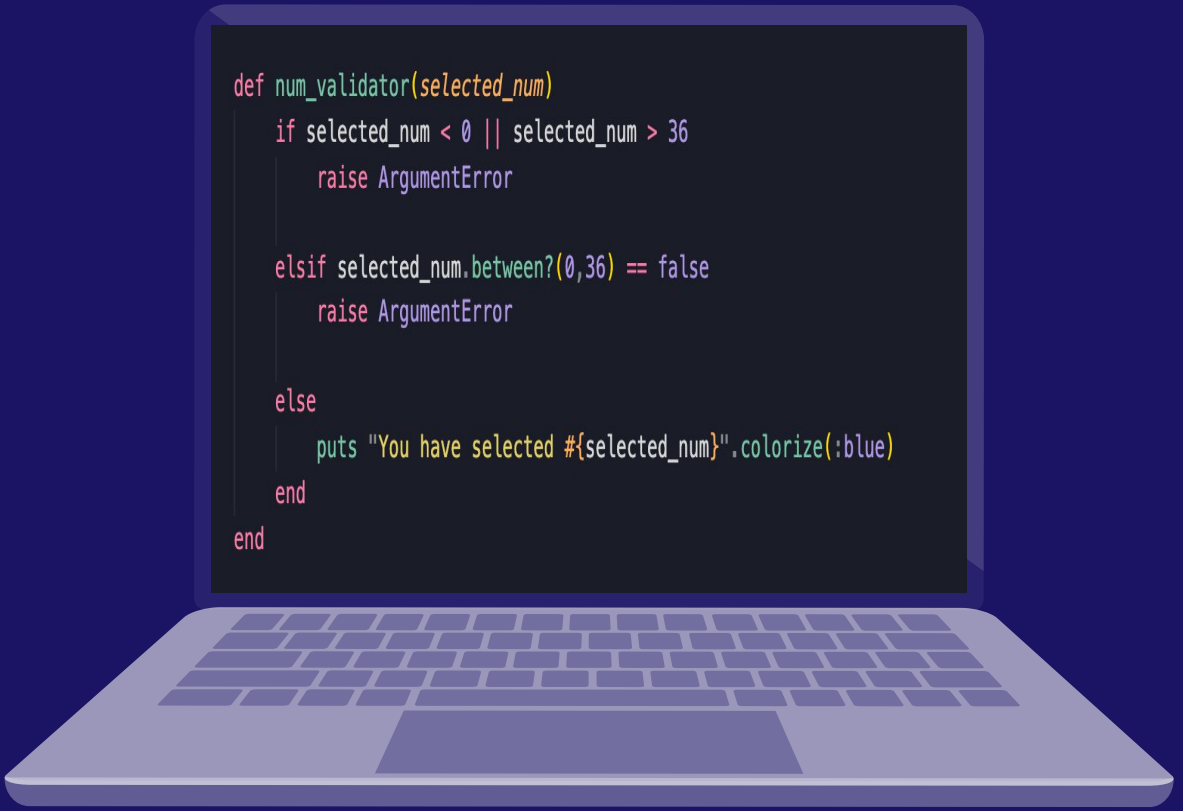
```
if input == 'Spin' && rand_num == selected_num
    credit += 20
    congrats
    credit_output(credit)
elsif input == 'Spin' && rand_num != selected_num
    credit = (credit.to_i) - 5
    unlucky
    credit_output(credit)
elsif
        input == 'Range' && rand_num.between?(0,12) == true && selected_num.between?(0,12)
        credit += 5
        congrats
        credit_output(credit)
    elsif
        input == 'Range' &&  rand_num.between?(13,24) == true && selected_num.between?(13,24)
        credit += 5
        congrats
        credit_output(credit)
    elsif
        input == 'Range' &&  rand_num.between?(25,36) == true && selected_num.between?(25,36)
        credit += 5
        congrats
        credit_output(credit)
    elsif
        input == 'Range' && rand_num != selected_num && rand_num.between?(0,36) == true && selected_num.between?(0,36)
        credit = (credit.to_i) - 5
        unlucky
        credit_output(credit)
    else
        "Not a valid input, please try again"

end

end
```

Code Overview

Control Flows:
Utilising if/else statements to validate if user's number is in line with the randomly generated number



```ruby
def num_validator(selected_num)
    if selected_num < 0 || selected_num > 36
        raise ArgumentError

    elsif selected_num.between?(0,36) == false
        raise ArgumentError

    else
        puts "You have selected #{selected_num}".colorize(:blue)
    end
end
```

Code Overview

Methods:
Utilising methods to raise ArgumentErrors if user enters incorrect number ie. (a number outside of 0-36)

```ruby
#Asks user to input their name, and stores into name variable
puts a.asciify("Welcome")
puts a.asciify("to")
puts a.asciify("Roulette")
begin
puts "What is your name?".colorize(:blue)
name = gets.chomp
validate_name(name)
puts "Hello there, #{name}!".colorize(:yellow)
rescue InvalidNameError
    puts "Please enter a valid name"
    retry
end


#Asks user how many credits they would like to play with
credit_select = prompt.select("How many credits would you like to begin with?", %w(20 50 100)).to_i
credit = credit_select.to_i

#Run credits method
credits(credit)


loop do

    if credit < 5
        puts "Sorry you're out of credit, Goodbye..."
        fancy_line
        break
    end
```

Code Overview

- Utilising gems to prettify code.
- Loops to break(end) the program once the user runs out of credit.
- Gets method to ask for user input.
- Type coercion to convert input data type into an integer.

Development Process

Step 1

Read assignment requirements.

Step 2

Scope out main features and functionality and purpose of App

Step 3

Trello to plan out tasks and features and order by priority. Use kanban style board to keep track of need to do items in real time.

Step 4

Create a repo for my project on github, and start to write the code.

Step 5

Use git commit regularly to keep of track milestones.

Step 6

Test code constantly to see if it is working as expected.

Development Process(cont)

Building a Terminal App

Challenges

Working with methods and variable scopes, getting the flow of program correct(different parts of code interacting in the correct flow)

Ethical issues

Gambling being a real problem in Australia - app is meant to serve as entertainment, but also as a tool to raise awareness of how easy it to lose and the game isnt always in the users control.

Favourite Parts

Writing code and testing and making it work. Researching and learning new functions, methods and ways to do things. Troubleshooting and debugging.

Tools

Git & Github, VS Code, Trello, RubyGems.org, Draw.io(FlowChart)

The End