Overview

Planet Finder is a terminal-based tool to search through planet data stored in Json files.

Setup

Github repo

1. Place all files in a dedicated folder

required files:

- planetfinder_ui.py
- planetfinder_find.py
- planetfinder_functions.py
- planetfinder_config.py
- planetfinder_visuals.py
- core.json

2. Run the planetfinder_ui.py program
3. Enter '2'
4. Enter a label for the data (for example 'core')
5. Copy and paste the file path of the core.json file
6. Make sure it's loaded by entering '3' on the main menu to show data

Modules

planetfinder_config (class Config) public methods:

- __init__()
  Creates a Config object and initializes path, data,paths, and region_data.

- ensure_config_exists()
  If the config.json file is not yet created or isn't where it should be this method will create a new config.json file with the default values.

- update_config()
  Updates the config.json file to contain what the Config object has in its path and data_paths .

- load_config()
  Opens the config.json file and sets the Config object's path to what the config.json has for the path key.
  Will also iterate through and read the files with the paths in the config.json "region_data_path(s)". The result being a dictionary containing the hierarchical dictionary data of each file listed in the config.json.

- update_region_data(region_label, data_path)
  Opens the file with data_path and reads in its data into the Config object's region_data and gives it the label 'region_label'.

planetfinder_visuals

- _get_char()
  Reads individual character inputs entered by the user.

- color_input(prompt, color)
  Uses _get_char() to repeatedly read character inputs and print them out with color.

- clear_console():
  Resets the cursor to the top left effectively clear the text in the terminal.

planetfinder_functions

- load_star_data(config)
  Prompts for the user to input a label and file path for the Json file containing the data.

- show_data(config)
  Prints out the data summary which includes number of systems, planets, and moons.

- show_config(config)
  Prints out each of the values in the config.json file.

- end_program(config)
  Ends the program.

- list_prompts()
  Prints out the available commands to the user.

- verify_prompt_choice(choice)
  Returns true if choice is a valid option in the list of commands.

planetfinder_find

- planet_options()
  Prints out all the available planet parameters the user can choose.

- view_parameters(parameters)
  Prints out the current planet parameters that the user has chosen.

- planet_prompt()
  Prints out serveral input options for the user.

- planet_step(config, parameters)
  Combines planet_options(), view_parameters(), and planet_prompt() into one step.

- planet_add_parameter(parameters, choice)
  Adds a parameter tuple to the parameters linked list.

- choice_action(choice, parameters)
  Depending on the user's choice this function could do one of the following:
  1. begin ending sequence to print out the ID list
  2. Exit to main menu without printing the id list
  3. Undo last choice
  4. Redo last undone
  5. Add choice to parameters
  6. Prints "invalid choice"

- search(config, planet_parameters)
  Iterates through the data and appends IDs which fit the parameters then returns the ID list.

- find(config)
  Combines the planet_step() and search() functions into one step.

planetfinder_find (class Node)
public methods:

- __init__(data)
  Creates a Node object with self.data = data, self.next = None, self.prev = None.

planetfinder_find (class LinkedList) public methods:

- __init__()
  Creates a LinkedList object with self.first = None, self.focus = None.

- append(data)
  Appends the data to the end of the LinkedList.

- next()
  Moves self.focus to the next Node in the LinkedList.

- prev()
  Moves self.focus to the previous Node in the LinkedList.

- solidify()
  Cuts off all the Nodes after self.focus.

planetfinder_ui

- setup_config()
  initializes the Config object and ensures it's ready to use.

- command_prompt_ui(config)
  The main outer loop for the main menu.

- main()
  Where everything starts.