class CPU

Constant Class Attributes

MAX = 9999	Sets the maximum sized "valid" instruction
MIN = -9999	Sets the minimum sized "valid" instruction
ACCUMULATOR_DEFAULT = 0000	Sets the Default Accumulator value when booting up
MAX_INSTRUCTION_LIMIT = 100	Forces the CPU to Stop when running more than 100 consecutive commands. Is configurable but useful for testing

Attributes

Accumulator : Float	Holds the arthimatic value between instructions Persists between instructions
Register : Int	Variable used for all instructions to be read from, or any data to be loaded. Does *not* persist between instructions
Pointer : Int	Variable used to hold the address that will read the next instruction from
Halted : Boolean	Variable used to see if the CPU has halted
Memory : Memory Object	Variable used to reference the Memory object

Methods

boot_up()
Used to reset the attributes to the constant default values

run()
Used to 'start' the CPU and will continuously read/run instructions and update the pointer until a halt command is hit

Raises ValueError : If operation() method cannot run the instruction given to it
Raises ValueError : If Memory cannot read from the pointer's address a valid instruction
Raises Halt : If Halt instruction is ran

decypher_instruction(word)
Splits the instruction into the first two digits and last digits (operator/operand)

Return : Returns a tuple of the operator and operand. If a 4-digit instruction is given, the operator will be the first 2 digits and the operand will be the last 2 digits

Raises ValueError : If a negative operation is attempted to split
Raises ValueError : If operation is not in valid range (not 4 digits or under 1000)

operation(word)
Runs the decypher_instruction() method
Takes the operator/operand combo and runs the operator through a match switch. Then runs the correct operation with the corect operand

Raises Halt : If Halt instruction is given
Raises ValueError : If operator that is given does not correspond with a valid operation

read_from_memory(addresss)
Used for access to the Memory class. Uses memory to set the value at an address into the register

load_to_memory(address, value)
Transition method to put data in the register into memory

op_READ(operand, gui)
Reads from the gui given to get a value.
Places that value into the memory adress given by the operand

op_WRITE(operand, gui)
Gets value from the memory address given by the opearnd
Prints value to the screen via the gui

op_LOAD(operand)
Loads the value at the address/operand location into the accumulator

op_STORE(operand)
Loads the value of the accumulator into the address/operand location in memory

op_ADD(operand)
Loads the value at the address/operand location into the register
Adds the register and the accumulator and leaves value in the accumulator

op_SUBTRACT(operand)
Loads the value at the address/operand location into the register
Subtracts the register from the accumulator and leaves value in the accumulator

op_MULTIPLY(operand)
Loads the value at the address/operand location into the register
Multiplies the register and the accumulator and leaves value in the accumulator

op_DIVIDE(operand)
Loads the value at the address/operand location into the register
Divides the register from the accumulator and leaves value in the accumulator

op_BRANCH(operand)
Moves the pointer to the given value as the operand

op_BRANCHNEG(operand)
Moves the pointer to the given value as the operand IF the accumulator is currently negative

op_BRANCHZERO(operand)
Moves the pointer to the given value as the operand IF the accumulator is currently ZERO

op_HALT()
Raises Halt
This cascades up into the run() method which then ends the CPU

class Memory

Attributes

size : Int
The total amount of memory cells that can be accessed

memory : Array of String
The reference to the array of all the memory cells

Methods

validate_address(address)
Validates whether an address is within bounds or not

Raises IndexError : If address is out of bounds

@staticmethod
validate_word(word)
static method used for determining if a word is valid to be inputted into memory

Raises ValueError : If the length of the word is NOT 5 - Must be 4-digits and +/- sign

Raises ValueError : If the first character of the word is neighter "+" or "-"

Raises ValueError : If the other 4 characters are not digits

Raises ValueError : If the word is not considered a string

read(address)
Used for the CPU to read a specific value from a specific address
Address is validated before performing using validate_address() method

write(address, word)
Used for the CPU to write a specific value to a specific addres
Address is validated before performing using validate_address() method
Word is validated before performing using validate(word)

clear()
Resets all memory cells to "+0000"

__str__()

Creates a single string that shows the entire memory and all its values

word_to_int(word)
Static method : used for converting a word to an integer when passing between CPU and Memory

int_to_word(number)
Static method : Used for converting an integer to a word when passing between Memory and CPU

Raises ValueError if Number isn't in range (-9999 to 9999)

class Bootstrapper

Attributes

Memory : Memory Object
CPU : CPU Object

Methods

load_program(program) : Loads a list of instructions into the memory module
Opens the file at the given file_name

Raises IndexError : If a file is too large, and too much memory is tried to be written, only the first 100 lines are accepted into memory and returns successfully

Raises ValueError : If an incorect instruction is read, instead of failing the value "+0000" is written to memory instead and then continues on

load_from_file(file_name) : Loads a file and then calls load_proram method

run(gui) : Method that starts the internal CPU with the given memory

class App

Attributes

boot : Bootstrapper Object

mem : Memory Object

cpu : CPU Object

Methods

setup_root() : sets up the root behavior window

setup_menu_bar() : Sets the menu bar and its behavior

setup_program_frame() : Sets up the program frame, the area for loading in and editing the program

check_text_length(event) : Binded function to the program frame, that on keydown event will check if valid length in program. If invalid, will reset text to before event

setup_main_frame() : Sets up the main frame of the program, including all child frames

setup_instruction_frame(main_frame) : Sets up the insrution frame for providing information to the user

setup_memory_frame(main_frame) : sets up the memory frame containing the memory info

setup_control_frame(main_frame) : sets up the control frame for controlling the program

save_file(file_path : str) : Save the contents of the program_text widgit to a flile

load_file(file_path : str) Load a file into the program_text widget

clear_program() : clear the program_text widget

load_memory() : Load te program_text widget contents into memory

adjust_memory_font_size() : Dynamically adjust font size to fit text within memory_text widget with some padding

update_memory_text() : update the memory_text widget with the current memory contents and highlights the pc

run_program() : runs the program

step_program() : Takes an incremental step in the CPU, performs one instruction and displays state

reset_program() : Reset the program and reset labels and text widgets to default values

instruction_window() : Display the instructions set window

open_color_dialog() : Open the color choose dialog

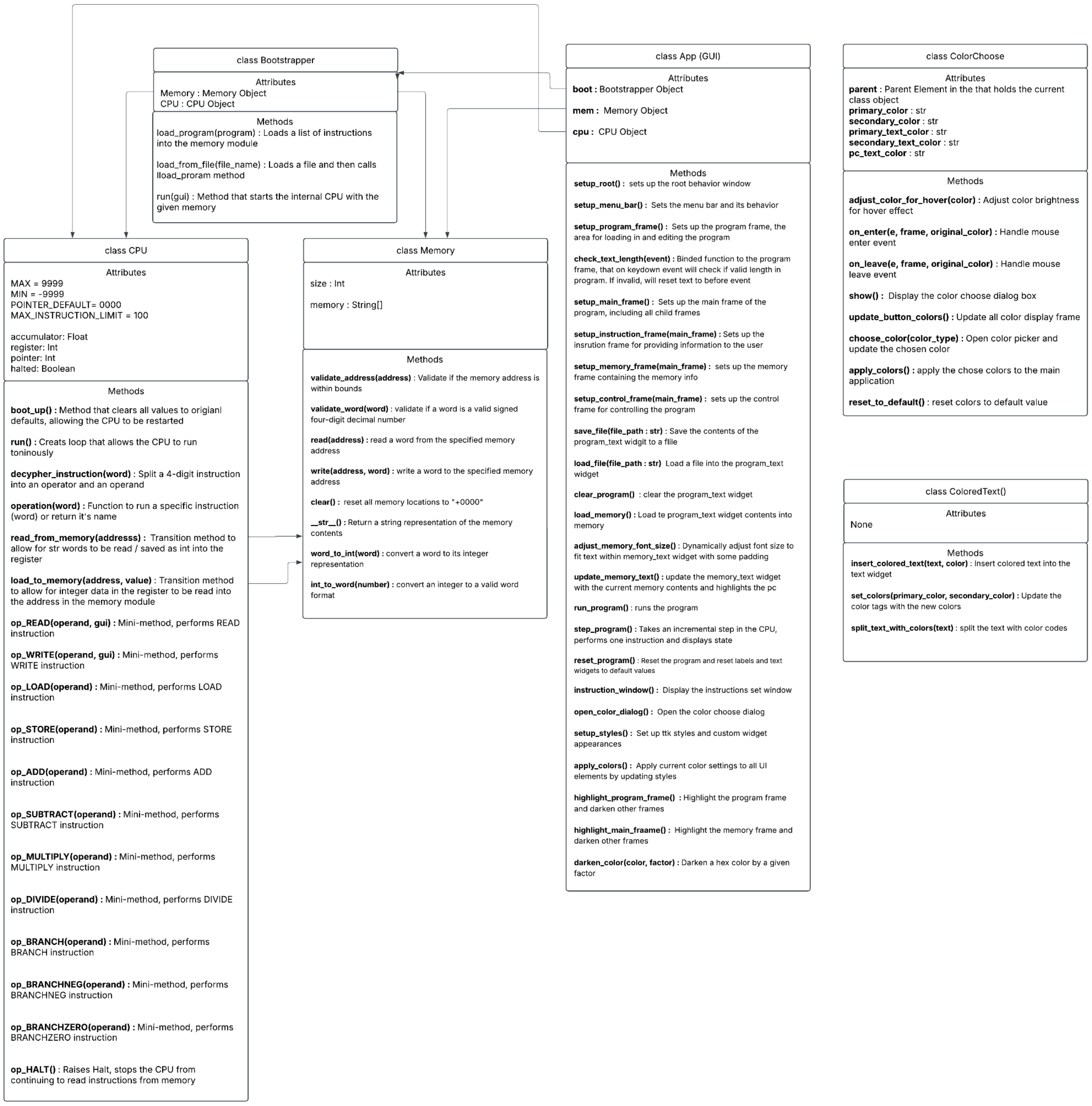
setup_styles() : Set up ttk styles and custom widget appearances

apply_colors() : Apply current color settings to all UI elements by updating styles

highlight_program_frame() : Highlight the program frame and darken other frames

highlight_main_fraame() : Highlight the memory frame and darken other frames

darken_color(color, factor) : Darken a hex color by a given factor



class Bootstrapper

Attributes

Memory : Memory Object
CPU : CPU Object

Methods

load_program(program) : Loads a list of instructions into the memory module

load_from_file(file_name) : Loads a file and then calls lload_proram method

run(gui) : Method that starts the internal CPU with the given memory

class App (GUI)

Attributes

boot : Bootstrapper Object

mem : Memory Object

cpu : CPU Object

Methods

setup_root() : sets up the root behavior window

setup_menu_bar() : Sets the menu bar and its behavior

setup_program_frame() : Sets up the program frame, the area for loading in and editing the program

check_text_length(event) : Binded function to the program frame, that on keydown event will check if valid length in program. If invalid, will reset text to before event

setup_main_frame() : Sets up the main frame of the program, including all child frames

setup_instruction_frame(main_frame) : Sets up the insrution frame for providing information to the user

setup_memory_frame(main_frame) : sets up the memory frame containing the memory info

setup_control_frame(main_frame) : sets up the control frame for controlling the program

save_file(file_path : str) : Save the contents of the program_text widgit to a flile

load_file(file_path : str) Load a file into the program_text widget

clear_program() : clear the program_text widget

load_memory() : Load te program_text widget contents into memory

adjust_memory_font_size() : Dynamically adjust font size to fit text within memory_text widget with some padding

update_memory_text() : update the memory_text widget with the current memory contents and highlights the pc

run_program() : runs the program

step_program() : Takes an incremental step in the CPU, performs one instruction and displays state

reset_program() : Reset the program and reset labels and text widgets to default values

instruction_window() : Display the instructions set window

open_color_dialog() : Open the color choose dialog

setup_styles() : Set up ttk styles and custom widget appearances

apply_colors() : Apply current color settings to all UI elements by updating styles

highlight_program_frame() : Highlight the program frame and darken other frames

highlight_main_fraame() : Highlight the memory frame and darken other frames

darken_color(color, factor) : Darken a hex color by a given factor

class ColorChoose

Attributes

parent : Parent Element in the that holds the current class object
primary_color : str
secondary_color : str
primary_text_color : str
secondary_text_color : str
pc_text_color : str

Methods

adjust_color_for_hover(color) : Adjust color brightness for hover effect

on_enter(e, frame, original_color) : Handle mouse enter event

on_leave(e, frame, original_color) : Handle mouse leave event

show() : Display the color choose dialog box

update_button_colors() : Update all color display frame

choose_color(color_type) : Open color picker and update the chosen color

apply_colors() : apply the chose colors to the main application

reset_to_default() : reset colors to default value

class CPU

Attributes

MAX = 9999
MIN = -9999
POINTER_DEFAULT= 0000
MAX_INSTRUCTION_LIMIT = 100

accumulator: Float
register: Int
pointer: Int
halted: Boolean

Methods

boot_up() : Method that clears all values to origianl defaults, allowing the CPU to be restarted

run() : Creats loop that allows the CPU to run toninously

decypher_instruction(word) : Split a 4-digit instruction into an operator and an operand

operation(word) : Function to run a specific instruction (word) or return it's name

read_from_memory(addresss) : Transition method to allow for str words to be read / saved as int into the register

load_to_memory(address, value) : Transition method to allow for integer data in the register to be read into the address in the memory module

op_READ(operand, gui) : Mini-method, performs READ instruction

op_WRITE(operand, gui) : Mini-method, performs WRITE instruction

op_LOAD(operand) : Mini-method, performs LOAD instruction

op_STORE(operand) : Mini-method, performs STORE instruction

op_ADD(operand) : Mini-method, performs ADD instruction

op_SUBTRACT(operand) : Mini-method, performs SUBTRACT instruction

op_MULTIPLY(operand) : Mini-method, performs MULTIPLY instruction

op_DIVIDE(operand) : Mini-method, performs DIVIDE instruction

op_BRANCH(operand) : Mini-method, performs BRANCH instruction

op_BRANCHNEG(operand) : Mini-method, performs BRANCHNEG instruction

op_BRANCHZERO(operand) : Mini-method, performs BRANCHZERO instruction

op_HALT() : Raises Halt, stops the CPU from continuing to read instructions from memory

class Memory

Attributes

size : Int

memory : String[]

Methods

validate_address(address) : Validate if the memory address is within bounds

validate_word(word) : validate if a word is a valid signed four-digit decimal number

read(address) : read a word from the specified memory address

write(address, word) : write a word to the specified memory address

clear() : reset all memory locations to "+0000"

__str__() : Return a string representation of the memory contents

word_to_int(word) : convert a word to its integer representation

int_to_word(number) : convert an integer to a valid word format

class ColoredText()

Attributes

None

Methods

insert_colored_text(text, color) : Insert colored text into the text widget

set_colors(primary_color, secondary_color) : Update the color tags with the new colors

split_text_with_colors(text) : split the text with color codes