Will Power-Jenkins
INF1340
Thursday, November 18th, 2021

Cryptarithm API Documentation

Program Overview:

Section 1: These first functions convert one of the given puzzles from letters to numbers in the form of an equation and provides the necessary code to solve the cryptarithm selected in section 2. This section was created by Peter Norvig in 2014* in order to solve cryptarithms in Python and has been modified slightly to fit the parameters of this program.

Section 2: This section stores the collection of cryptarithm puzzles and then randomly selects one to be solved by the `solve` function created in the preceding section

Section 3: This section of the program creates the GUI that the user will be interacting with using Tkinter, and generates a certain number of entry boxes based on the characters in the puzzle as determined by the `letter_replace`, `valid`, and `solve` functions.

Note: Functions are listed in alphabetical order

Functions:

__init__ (sample)

```
def __init__(self):
      window = Tk() # Create a window
      window.title("Check Cryptarithm Solution") # Set title
      colour_main = "papaya whip" #make the window look pretty
      window.configure(bg = colour_main)
      window. geometry("375x400")
      puzzle_font = Font(family="Arial Narrow", size=12, weight="bold")
      …
      prompt = Message(window, text="Enter the solution for the cryptarithm
      displayed here:", bg=colour_main, justify=CENTER, font=prompt_font)
      prompt.grid(row=1, column=1)
      …
      for i in range(len(puzzle_list)):
                  Label(window, text=(puzzle_list[i], "="), bg=colour_main)
                  .grid(row = i+2, column=2)
                  user_input = Entry(window, width=3, justify=CENTER,
                  relief="ridge")
      …
```

*Norvig, P. (2014). Cryptarithmetic (alphametic) problems. *GitHub*.
https://github.com/norvig/pytudes/blob/main/ipynb/Cryptarithmetic.ipynb

- This function is the main tkinter function that creates the GUI for the user to interact with. The verify_values function operates within it, and arguments passed within it include `Tk()` (which creates a window), `.title` (labels the GUI), `.geometry` (sets the size of the window, `Font()` (sets the font face and size, etc.), Entry() (generates an input box for the user the input information), as well as `Message()` and `Label()` (generate text to convey information to the user).

letter_replace:

```
def letter_replace(formula):
      formula = formula.replace(' = ', ' == ')
   letters = cat(set(re.findall('[A-Z]', formula)))
      for digits in itertools.permutations('1234567890', len(letters)):
         yield formula.translate(str.maketrans(letters, cat(digits)))
```

- The `letter_replace` function is the function that ensures all the letters in the selected cryptarithm are converted to the appropriate numbers. Using the .`replace`, `cat()`, `.permutations`, `len()`, .`translate`, and `.maketrans` functions and methods.
    - .`replace`: Ensures Python can recognize equivalent values mathematically in this case
    - `cat():` Concatenate
    - `.permutations:` *itertools* method that generates all known permutations of a given list
    - `len():` Counts length of variable – used in this case to find all the permutations of 0 through 9 with a length of the given puzzle
    - .`translate`: Returns a string with characters replaced by previously defined rule (as stored using the `maketrans()` function
    - `.maketrans:` Creates a mapping table for character replacement

solve:

```
def solve(formula):
      return filter(valid, letter_replace(formula))
```

- This function from section 1 takes both the `valid` and the `letter_replace` functions and filters them so as to be able to solve the given puzzle.

valid:

```
def valid(exp):
    try:
        return not lead_zero(exp) and eval(exp) is True
    except ArithmeticError:
        return False
```

- This function ensures that the returned results do not contain a leading zero (as that would make a cryptarithm unsolvable) and that the evaluated expression is true (by stopping the function if there is an arithmetic error.

verify_values:

```
def verify_values():
       entry_list = ""
           for entries in user_list:
                entry_list = entry_list + str(entries.get())
            entry_list = list(entry_list)
            if entry_list == solution_list:
                  tkinter.messagebox.showinfo("Check Cryptarithm Solution",
                  "Correct!")
            else:
                tkinter.messagebox.showerror("Check Cryptarithm Solution",
                "Incorrect. Please try again.")
```

- This function verifies that the user input generated list (converted into a Python list using the `list()` function) is equivalent to the list generated by the `solve` function. Depending on the validity of the entry, the user receives a message using the `.messagebox` method.