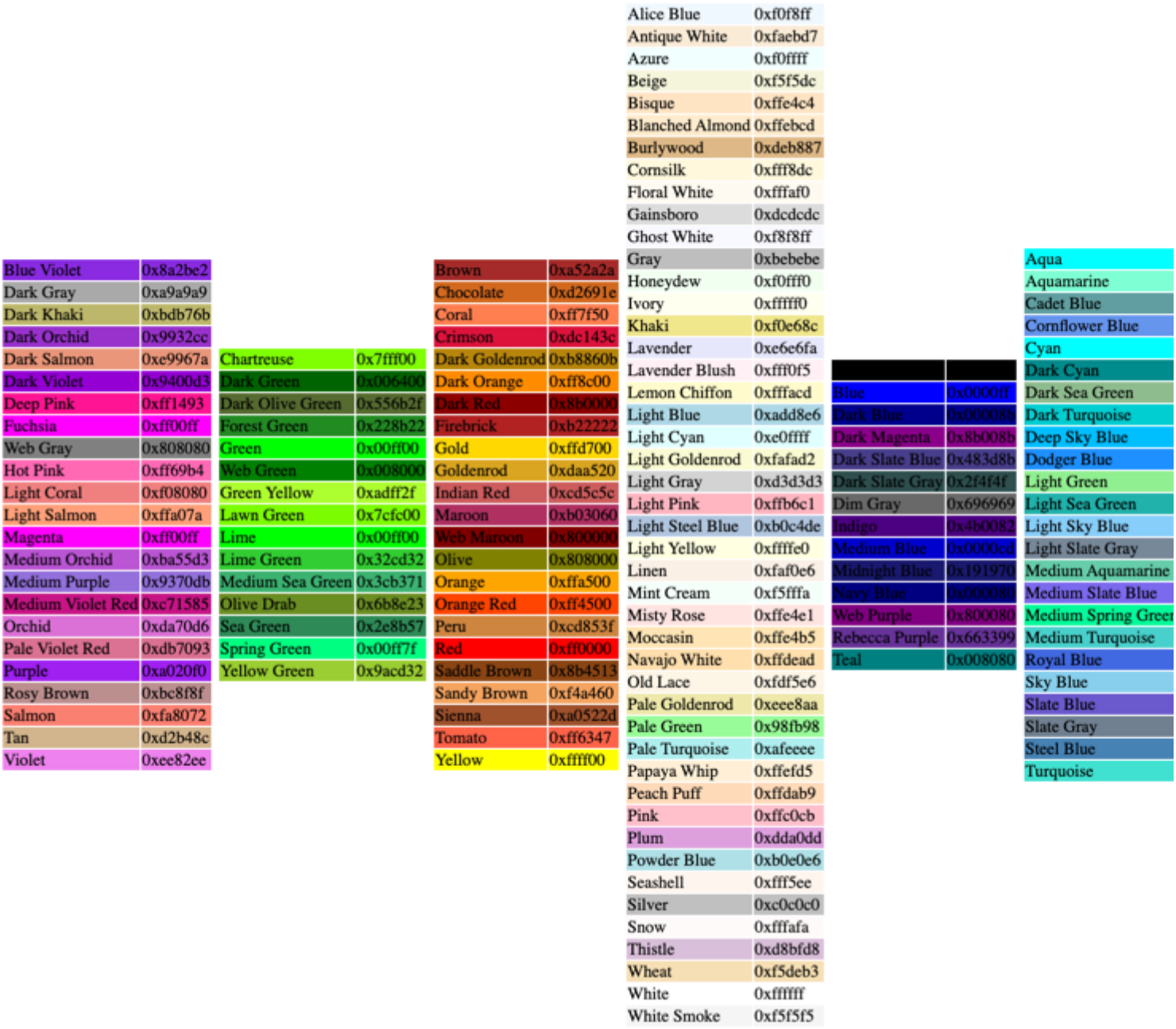
Color Survey



Results of a simple color categorization using k-means clustering in Python

[The graphic is the output of an assignment for a 2nd quarter class in Python. It's not related to this 3400 assignment.]

Summary

In this assignment, you will be tabulating results of a survey where people were asked to vote for their top three favorite colors (in order). The results of the survey are stored in an input file that has one or more lines in the following format:

```
blue green red
```

- Each line refers to one survey response.
- The first color listed is the favorite color (call this a 1st place vote). The second color listed is the second favorite color (call this a 2nd place vote). The third color listed is the third favorite color (call this a 3rd place vote).
- The colors are separated by a single space.
- Each color is a single word consisting of only lowercase letters.

Directions

You must create the following functions in the file *hw1.py*.

process_file

Parameters: Name of the input file (string)
Returns: A dictionary consisting of key-value pairs where the key is string and the value is a tuple consisting of three integers.
Description: Reads and parses the input file. Creates and returns a dictionary consisting of key-value pairs where the key is a color and the value is a tuple consisting of three integers: (number of 1st place votes, number of 2nd place votes, number of 3rd place votes). The dictionary only contains colors that appeared in the file.
Assumptions: The input file exists and is formatted as described above. The input file contains at least one line.

get_first_place_votes

Parameters: Dictionary returned from *process_file*, color (string)
Returns: Integer.
Description: Returns the number of 1st place votes (possibly zero) for the provided color.
Additional Restrictions: You must not use a loop in this function.

create_favorite_color_list

Parameters: Dictionary returned from *process_file*
Returns: A list of strings.
Description: Returns an ordered list of colors based on the number of 1st place votes. The first item in the list is the color that had the most 1st place votes. The second item in the list is the color that had the second most 1st place votes. The list only contains colors that receive at least one 1st place vote. Ties are broken as follows: 1) winner is the color with higher number of 2nd place votes, 2) if still tied, winner is the color with higher number of 3rd place votes, 3) if still tied, winner is the color that appears earlier in alphabetical order.

create_color_score_dict

Parameters: Dictionary returned from *process_file*
Returns: A dictionary consisting of key-value pairs where the key is a string and the value is an integer.

Description: Creates and returns a dictionary consisting of key-value pairs where the key is a color and the value is an integer that is computed using the following formula (number of 1st place votes x 3) + (number of 2nd place votes x 2) + (number of 3rd place votes). The dictionary only contains colors that appeared in the file.

print_dictionary

Parameters: Any dictionary
Returns: nothing
Description: Prints the dictionary in sorted order (use *sorted* function). Print each entry on a separate line in the following format:
key: value
Additional Restrictions: You can make no assumptions on the type of dictionary.

Top-Level Functionality

In addition to these functions, you must do the following steps to create a simple test driver. This code should reside outside the functions:

1. Get the name of an input file from the command line (using *sys.argv*). WARNING: Do not prompt the user for a file name.
2. Call *process_file* with the file name from step 1 storing its result.
3. Call *print_dictionary* to print the result from step 2 to the screen.
4. Call *get_first_place_votes* with *blue* storing its result.
5. Print the result from step 4 to the screen.
6. Call *get_first_place_votes* with *green* storing its result.
7. Print the result from step 6 to the screen.
8. Call *create_favorite_color_list* storing its result.
9. Print the result from step 8 to the screen.
10. Call *create_color_score_dict* storing its result.
11. Call *print_dictionary* to print the result from step 10 to the screen.

Sample Input File and Output

Assuming there is a sample color file as shown:

```
$ cat sample1.txt
red white blue
blue yellow green
blue purple pink
green yellow red
blue pink green
green purple black
red green blue
yellow pink brown
blue white black
purple pink green
green black red
red green white
```

```
purple red green
$ python3 hw1.py sample1.txt
black: (0, 1, 2)
blue: (4, 0, 2)
brown: (0, 0, 1)
green: (3, 2, 4)
pink: (0, 3, 1)
purple: (2, 2, 0)
red: (3, 1, 2)
white: (0, 2, 1)
yellow: (1, 2, 0)
4
3
['blue', 'green', 'red', 'purple', 'yellow']
black: 4
blue: 14
brown: 1
green: 17
pink: 7
purple: 10
red: 13
white: 5
yellow: 7
```

IMPORTANT! Do not assume that testing is complete if your program produces the correct output for this provided input file. During grading, your assignment will be tested with other input files.

Grading Notes

- A test that crashes due to a run-time exception will be considered a failing test.
- Programs that contain syntax errors will receive a zero.

Version

- Last updated 05-Jan-2022

HW1 Rubric

Criteria	Ratings					Pts
process_file	**10 pts** **Full Marks**	**8 pts** **Passes most tests**	**5 pts** **Passes some tests**	**2 pts** **Attempted**	**0 pts** **No Marks**	10 pts
get_first_place_votes	**6 pts** **Full Marks**	**5 pts** **Passes most tests**	**3 pts** **Passes some tests**	**1 pts** **Attempted**	**0 pts** **No Marks**	6 pts
create_favorite_color_list	**12 pts** **Full Marks**	**9 pts** **Passes most tests**	**6 pts** **Passes some tests**	**2 pts** **Attempted**	**0 pts** **No Marks**	12 pts
create_color_score_dict	**10 pts** **Full Marks**	**8 pts** **Passes most tests**	**5 pts** **Passes some tests**	**2 pts** **Attempted**	**0 pts** **No Marks**	10 pts
print_dictionary	**6 pts** **Full Marks**	**5 pts** **Passes most tests**	**3 pts** **Passes some tests**	**1 pts** **Attempted**	**0 pts** **No Marks**	6 pts
Other functionailty, style, and documentation Follows PEP-8 style guide. Has suitable docstrings for module and each function.	**6 to >0.0 pts** **Full Marks**			**0 pts** **No Marks**		6 pts
						Total Points: 50