```
"""
    Contributors: Jesus A. Bernal Lopez jebernal@csumb.edu
                  Paul Whipp pwhipp@csumb.edu

    Class: CST-205

    Lab 3: Color Dictionaries

    Last Modified: 02/02/2019

    Due Date: 02/04/2019
"""

# Dictionary declaration for task 1
color_dictionary = {
        "red": (255, 0, 0),
        "green": (0, 255, 0),
        "blue": (0, 0, 255),
        "magenta": (255, 0, 255),
        "cyan": (0, 255, 255),
        "yellow": (255, 255, 0),
        "purple": (92, 40, 136),
        "lime": (9, 234, 79),
        "black": (0, 0, 0),
        "white": (255, 255, 255),
        "lemon": (255, 252, 81),
        "gold": (206, 203, 10),
        "gray": (165, 165, 145),
    }

sentence_fragments = [
    "The red channel of",
    "The green channel of",
    "The blue channel of",
    ]

# I think "is" makes more sense than "has value" but hey I am not the instructor
has_value = "has value"

tineye_sample = {
    "status": "ok",
    "error": [],
    "method": "extract_collection_colors",
    "result": [
        {
            "color": (141, 125, 83),
            "weight": 76.37,
            "name": "Clay Creek",
            "rank": 1,
            "class": "Grey"
        },
        {
            "color": (35, 22, 19),
            "weight": 23.63,
```

```
            "name": "Seal Brown",
            "rank": 2,
            "class": "Black"
        }
    ]
}


def print_dictionary(dict):
    statement = "{\n"
    for key in dict:
        statement += f"\t{key}: {dict[key]},\n"
    statement += "}"
    print(statement)


"""
    Description: We got bored and decided to print the dictionary in a pretty way, the dictionary is declared up top
    Driver: Jesus
    Navigator: Paul
"""


def task1():
    print_dictionary(color_dictionary)


"""
    Description: We defined sentence fragment as a global list and we formed a sentence in the function asking for
                 the channel and color as arguments.
    Driver: Jesus
    Navigator: Paul
"""


def task2_1(channel, color):
    # channel values:
    # red = 0
    # green = 1
    # blue = 2
    print(f"{sentence_fragments[channel]} {color} {has_value} {color_dictionary[color][channel]}")


"""
    Description: We were debating if the desired output was all the values with the second letter being 'e' (task2_2)
                 or running the color through a check to see if the second letter was 'e' then outputting the whole
                 tuple if it was (task2_3) and just the channel if it was not.
    Driver: Paul
    Navigator: Jesus
```

```
"""


def task2_2(dict):
    for color in dict:
        if color[1] == 'e':
            print(f"{color} {has_value} {color_dictionary[color]}")


def task2_3(channel, color):
    if color[1] == 'e':
        print(f"The tuple of {color} {has_value} {color_dictionary[color]}")
    else:
        task2_1(channel, color)


"""
    Description: We approached it from a couple different angles.
        1. We printed out the desired result by just navigating the dictionary in a hard coded way.
        2. We made a more generalized function that allowed the function to take the name, channel and dictionary
            as arguments and output the result in a sentence as in task 2.
    Driver: Jesus
    Navigator: Paul
"""


def task3(name_color, channel, dictionary):
    ##########################################################################################
    # This will do exactly what you asked for but below is a more
    # general way if the dictionary was bigger and we did not know the location of it.
    ##########################################################################################
    hard_coded_way_1 = tineye_sample["result"][0]["color"][0]
    hard_coded_way_2 = tineye_sample["result"][1]["color"][2]
    print(hard_coded_way_1)
    print(hard_coded_way_2)

    # result = dictionary["result"]
    # for val in result:
    #     if val["name"] == name_color:
    #         print(f"{sentence_fragments[channel]} {name_color} {has_value} {val['color'][channel]}")


def main():
    # task1()

```

```
    # task2_1(2, 'magenta')
    # task2_1(1, 'yellow')
    # task2_1(0, 'cyan')

    # task2_2(color_dictionary)

    # task2_3(2, 'red')
    # task2_3(1, 'blue')
    # task2_3(2, 'lemon')

    task3("Don't matter", "just to lazy to make another without", "arguments")
    # task3("Clay Creek", 0, tineye_sample)
    # task3("Seal Brown", 2, tineye_sample)


if __name__ == "__main__":
    main()


"""
**********************************************************************
*                             Summary                                *
**********************************************************************

Overall we got good at navigating through dictionaries. A problem we faced was debating what was asked for in task 2
but ultimately we overcame that by implementing both solutions.


**********************************************************************
*                             Task 1                                 *
**********************************************************************


Function call: task1()
Output:
{
        red: (255, 0, 0),
        green: (0, 255, 0),
        blue: (0, 0, 255),
        magenta: (255, 0, 255),
        cyan: (0, 255, 255),
        yellow: (255, 255, 0),
        purple: (92, 40, 136),
        lime: (9, 234, 79),
        black: (0, 0, 0),
        white: (255, 255, 255),
        lemon: (255, 252, 81),
        gold: (206, 203, 10),
        gray: (165, 165, 145),
}


```

```
**********************************************************************
*                            Task 2                                  *
**********************************************************************

*********************************
*           Part 1              *
*********************************


Function calls: task2_1(2, 'magenta')
                task2_1(1, 'yellow')
                task2_1(0, 'cyan')
Output:
The blue channel of magenta has value 255
The green channel of yellow has value 255
The red channel of cyan has value 0


*********************************
*           Part 2              *
*********************************


Function call: task2_2(color_dictionary)
Output:
red has value (255, 0, 0)
yellow has value (255, 255, 0)
lemon has value (255, 252, 81)


*********************************
*           Part 3              *
*********************************


Function calls: task2_3(2, 'red')
                task2_3(1, 'blue')
                task2_3(2, 'lemon')
Output:
The tuple of red has value (255, 0, 0)
The green channel of blue has value 0
The tuple of lemon has value (255, 252, 81)


**********************************************************************
*                            Task 3                                  *
**********************************************************************


Function calls: task3("Clay Creek", 0, tineye_sample)
                task3("Seal Brown", 2, tineye_sample)
Output:
The red channel of Clay Creek has value 141
The blue channel of Seal Brown has value 19

```

```
Function call(hard coded way): task3("Don't matter", "just to lazy to
 make another without", "arguments")
141
19


"""
```