

MCS 260 Project Two : a database of colors due Monday 15 February at noon

The goal of this project is to use if-else statements, anydbm, and turtle graphics in a Python program to make a database of colors. If the program is called `colordb.py`, then a session at the command prompt `$` happens as follows:

```
$ python colordb.py
hit enter to give a color code
or else type the name of a color :
-> give a triplet (*,*,*), * in 0..255 : (255,255,0)
-> give a name for the color (255, 255, 0) : yellow
-> added (yellow,16776960) to the color database
hit enter to exit
```

After the user enters the code `(255,255,0)`, a turtle window comes up with its *background* color corresponding to the given color code. After `yellow` is entered, the *title* of the turtle window changes into `yellow`. Note: $255 \times 256^2 + 255 \times 256 + 0 = 16776960$.

Another result of this session is that a file `colors.dbm` was made with `anydbm`. This file stores the map between color codes and color names. The color is stored twice: once with as key the name of the color and as value the code of the color; and once with as key the code of the color and as value the name of the color. Two followup sessions are

```
$ python colordb.py
hit enter to give a color code
or else type the name of a color : yellow
hit enter to exit
$ python colordb.py
hit enter to give a color code
or else type the name of a color :
-> give a triplet (*,*,*), * in 0..255 : (255,255,0)
hit enter to exit
```

In both followup sessions, a turtle window pops up with title `yellow` and background color with code `(255,255,0)`. In the first session, the code `(255,255,0)` is computed from `16776960`. In the second session, the code `(255,255,0)` is transformed into `16776960`.

To add a new color to the database, we could start entering the code (as done above with `(255,255,0)`) or start entering the name of the color. For example:

```
$ python colordb.py
hit enter to give a color code
or else type the name of a color : blue
-> "blue" is not in the color database
-> give a triplet (*,*,*), * in 0..255, for the color "blue" : (0,0,255)
-> added (blue,255) to the color database
hit enter to exit
```

Also in this case, a turtle window pops up after the user enters the code (0,0,255) with background color corresponding to the code and with window title equal to the given name.

For this project we use the integer color codes, via triplets of integer numbers between 0 and 255 (we have exactly $256^3 = 16,777,216$ different colors to choose from). Some useful turtle methods are illustrated below:

```
>>> import turtle
>>> turtle.colormode(255)      # switch to 0..255 ranges in triplets
>>> turtle.bgcolor((255,0,0)) # set the background color of screen
>>> turtle.title("red")       # red appears in title bar of window
```

As to `anydbm`, the types of keys and values are always strings, so observe the proper use of the `str()` and `eval()` commands. To every color triplet (c_0, c_1, c_2) corresponds a unique number $n = c_0 256^2 + c_1 256 + c_2$ and for every number in the range $0 \dots 16777215$, there is a unique color triplet.

Some important points:

1. You may assume the input of the user is always correct.
2. Keep the same formats of the dialogue with the user as shown in the sample sessions. In particular, stick to the order of events as specified. To be a relevant program, for a given new color code, the user must first be able to see the color appear in the turtle window before prompted to give a name to the color.
3. The first line of your Python program must be

```
# MCS 260 Project Two by <Author>
```

where you replace the `<Author>` by your name.
4. Add documentation to clarify your choice of variables and to indicate the steps of the program.
5. Handing in an incomplete but working program is better than handing in a program that crashes or does not run at all. In writing the code, you could first focus on the visualization with turtle graphics and then work on the persistency of data with `anydbm`, or otherwise first use `anydbm` and then do the graphics.
6. The computer project must be solved *individually*. Collaborations are not allowed.
7. Email your solution to the project to `jan@math.uic.edu` before noon on Monday 15 February so the date of the email is proof of an on time submission. Bring also a printed version of your solution to class.

If you have questions or difficulties with the project, feel free to come to my office for help.