ICS 212 Homework #3

In this assignment, further extend the gumball machine program which you have been developing in homework #1 and #2.

● First, eliminate the global variable which you have been using to contain a description of the gumball machine. As mentioned, previously, using global variables like this is usually verboten, and the only reason we did this in the previous assignment was that we had not yet covered the use of pointers to structures as function parameters. In this assignment, we will use the preferred approach of pointers to structures. To facilitate this, replace the global structure, with a local pointer to a gumball machine, which should be located inside function OperateGumBallMachine().

● Since you have eliminated the global variable, you will need to modify the functions which access the global variable, to take a pointer parameter. You will recall that they should receive a *pointer* to a structure as a parameter, rather than a structure as a parameter, because we want them to be able to alter the structure (e.g. function DeliverGumBall() needs to be able to decrement the gumball count once it has delivered a gumball, and RefillGumBallMachine() needs to be able to add to the gumball count in connection with its job). You will also need to use the -> syntax to access the fields of the structure, rather than the . syntax.

● Next, create a constructor function, to *dynamically allocate* a gumball machine structure, and initialize its values. You can name the constructor function whatever you'd like, but as always, please use a meaningful identifier. Here is where it will get a little tricky. Make it so that the actual colors of gumballs in the gumball machine are chosen randomly, at the time the gumball machine is constructed (and also when it is refilled). You should have a 'bunch' of colors to choose from (you can decide what a 'bunch' means, but it should be a pleasant selection), and a given constructed gumball machine should possess a relatively small subset of the total set of colors. You will obviously have to alter the original GumBallMachineType struct in order to accommodate this change. For debugging purposes, have the constructor print a message indicating how many and what color gumballs are in the gumball machine. You can make it so that the possible number of *colors* of gumballs per gumball machine is fixed, or variable – it is your choice (it depends upon how daring you feel).

●Finally, modify OperateGumBallMachine() so that it instantiates three gumball machines at once. In the main loop, before asking for coins, the function should ask the user which gumball machine to put coins into, and then that's the machine which the user should interact with during that iteration of the loop.

Have fun!