ECE118 LAB SEVEN

This week you are going to be creating backgrounds for simple cartoon animations - random cityscapes and happy street scenes. More than ever, success here depends upon a step-by-step divide and conquer approach. Simple little functions that act together to produce complex results.

After part 2d, the details of the picture will be randomly produced. Run your programs a few times, the results should be significantly different each time.

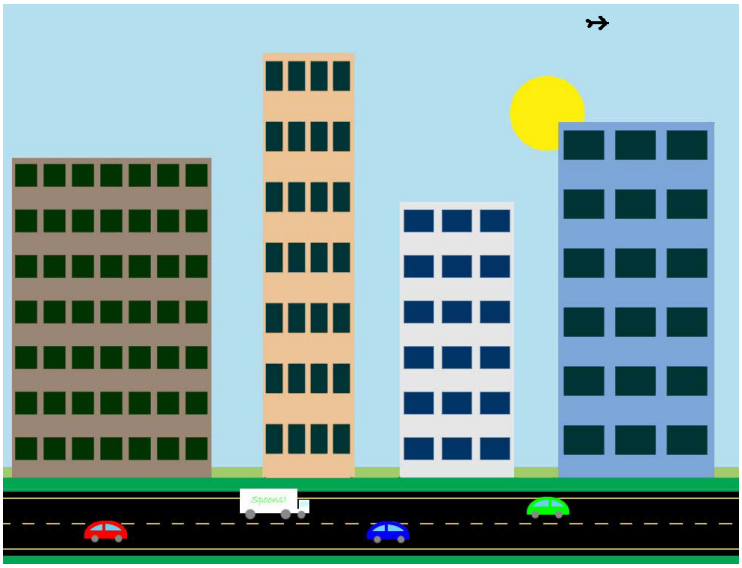
1. *Background*

Make your program randomly choose a time of day for the scene (remember the `random_in_range` function). Shade the background in an appropriate sky colour and put a sun or a moon or something like that up there. All the buildings and things will be drawn on top on this.

2. *Office Buildings*

Think of a modern dull office building. From any distance it is just a bland rectangle with a few rows and columns of blank windows.

Once you have created buildings you can add images of birds or aircrafts in the sky to make it more realistic.



Think about how you could produce a randomly generated cityscape like that. A series of simple steps gradually creeping up on a complete solution should work out quite well:

a. *Draw a single window*.

Define a function that is given the height, width, x and y position, and colour of a window, and draws a filled rectangle exactly as described.

b. *Draw a row of windows to make a single floor*.

Define a function that is given the number of windows required, the x position of where to start, and the height, width, y position, and colour desired for each individual window. It should of course draw the row of windows representing a whole floor.

c. *Draw a whole block of windows*.

Define a function that is given the number of floors required, the number of windows on each floor, the x and y position of where to start, and the height, width, and colour desired for each individual window. It should then draw the whole array of windows in a regular grid.

d. *Draw a whole office building*.

Define a function that is given the x and y position of the bottom left corner of an office block, and its desired width.

It should come up with random numbers in reasonable ranges to choose the height of the building, how many floors, how many offices per floor, the colour of the building itself, and the colour of its windows. You will need to experiment to come up with good ranges for these random numbers.

e. *Urban Infrastructure*.

A road is just some stripes put in the right place. Some modern cars are barely more than semicircles with windows and wheels. They don't even have to look good. Even South Park quality is much more than we expect. Just make a bit of an effort.

f. *Draw a city scene*.

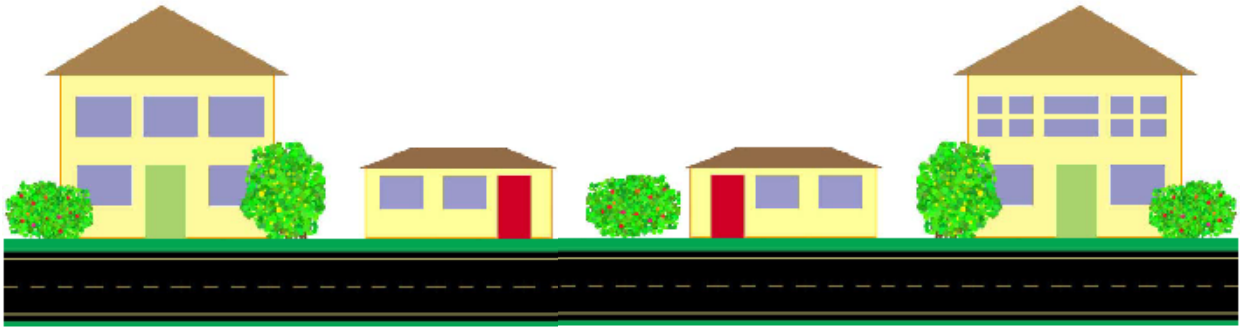
Define a function that is given the x and y position of the bottom left corner of a row of offices. It should draw one office block with a reasonably chosen random width, then if there is enough distance left before the right edge of the screen, it should continue to draw the rest of the row.

Important:

Take care that your office buildings don't collide with each other. It damages the brickwork, frightens the workers, and looks as though you weren't trying.

3. *Houses.*

A house is rather like a little office block. They tend to be restricted to one or two floors, and a modest number of windows on each floor, and they tend to have less garish colour schemes than offices, but a flat-roofed house should be easy to produce just by picking some of the random numbers differently.



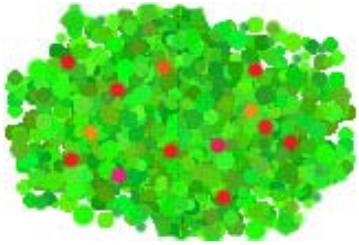
Of course, we want something better than that. Once the position and size of a house is known, it is easy to draw a roof on top of it. But what about the front door? You can probably think of a good technique - when the bottom floor is being drawn, one of the windows can be randomly replaced with something slightly different to represent a door.

We want some architectural variety, so your program must randomly choose whether each house has one or two storeys.

Make your city scene contain a random mix of houses and offices., Adding traffic lights and pedestrian crossings will make it look more natural.

4. *We Want A Shrubbery.*

A bush that doesn't look completely terrible is quite easy to draw: randomly sized dots of random shades of green constrained within a particular area (x and y chosen randomly within a range). Occasionally throwing in another colour might look like flowers or tomatoes or something.



Mine is not the nicest bush I've ever seen, and I'm sure you can do much better with a little bit of experimentation. The point is, we want a nice eco-friendly town, so give your houses a random selection of bushes and general plant life.

5. *Trees Are Good Too.*

We've got to have trees of course. A tree could be as simple as a bush with a trunk, but try for something better. With very little effort you can make something almost reasonable looking.

There is plenty of scope for extra credit in this lab, and the lab guys like to reward creativity. But remember that good tidy programming is essential. Pretty trees won't grow from ugly code.

6. *Make it look nice.*

As you've got a separate simple little function just for drawing one window, perhaps occasionally at random, it could put something in a window, like a cat or some curtains or whatever.

When things are drawn quite small, they can be remarkably effective with very little detail. Put various things in your scene. Maybe some people in the street, some grass on the ground, an American flag on a pole, a cannon on a roof, an aeroplane up in the sky...