Table 1 shows 6 types of Izhikevich cell models, 3 that we have already created the code for. In this bonus lab, we will create code for the additional model types.

1. For each cell type, create a separate file that will define the class of the cell type:
    - fast_spiking_cell.py
    - low_threshold_spiking_cell.py
    - Late_spiking_cell.py

2. For these cells, you'll notice the equations are structured differently:
    - The fast spiking cell requires an additional, piece-wise equation for calculating the next value of u and has an additional parameter vb
    - The low-threshold spiking cell has more complicated expressions for checking if the cell has reached a spike peak and then updating v and u if it has reached the peak
    - The late spiking cell has an additional variable to represent the membrane potential in its dendrite over time, vd

    You will need to override the super class's simulation method with one unique to that subclass, so that you can customize the equations inside the for loop.

3. In the simulation methods for each of the new cell types, you can paste in the simulation code from the super class and then update. There are multiple strategies available for updating the equations. One strategy is:
    - Fast Spiking Cell: Look at the equation for updating u. Refer back to the equation for u in the fast spiking cell row of the Table. Notice the term U(v), which we do not currently have in the equation in that second line of the function. Let's substitute the reference to v[0,i] there with a new variable, Uv. Now, add three additional blank lines between the first and second lines of the method. We will add three lines of code here to define the value of Uv. Per the equation in Table 1 showing that U(v) is a piece-wise function, we should add something like the following:
      ```
      Uv = 0
      if v[0,i]>=-55:
          Uv = 0.025*(v[0,i] - vb)^3 4
      ```
      We also need to define the parameter vb. We can set it to -55.

    - Low Threshold Spiking Cell: we don't need to change how v and u are calculated, only the check to see if v is above the peak value and the substatements for the equations that are solved if it is. Edit the if statement so that, rather than just checking if the value of v is greater than or equal to a specific value, it checks if v is greater than or equal to `(40 - 0.1*u[0,i+1])`. Also edit the substatements that are executed if the condition is true, so that v is reset to `-53 + 0.04*u[0,i+1]` and `u[0,i+1]` is reset to the minimum of two values using the np.min() function (ex: `np.min(u[0,i+1]+20,670)`).

- Late Spiking Cell: this model is different from the others because it actually models a two-compartment cell, one with both a cell body and a dendrite. The potential of the dendrite can be different from the potential of the soma, but they influence each other. The potential of the dendrite is stored in variable vd in the equations shown in Table 1 and in variable vd in the Python code. We'll also need to add an extra expression to the equation that calculates the contribution of the dendritic potential to the soma and we'll need to add an extra equation that calculates the dendritic potential at each time step. But before the simulation code even starts, we'll need to define vd as a vector just like we previously defined v. The following code lines can be used to do this:

```
vd = v.copy()
for i in range(0,n-1):
    v[0,i+1]=v[0,i]+tau*(k*(v[0,i]-vr)*(v[0,i]-vt)+
           1.2*(vd[0,i] - v[0,i])-u[0,i]+I[0,i])/C
    u[0,i+1]=u[0,i]+tau*a*(b*(v[0,i]-vr)-u[0,i])
    if v[0,i+1]>=30:
           v[0,i]=-45
           u[0,i+1]=u[0,i+1]+100
    Vd[0,i+1] = vd[0,i] + tau*(0.01*(v[0,i] - vd[0,i]))
```

  Tip: if you copy and paste these lines into your script file, make sure to check that the indentations are correct before moving on to the next step.

4. Once you have finished creating the new cell classes, you can add code lines to the end of each file that check whether the code is running from within the main scope and, if it is, creating a cell object, simulating it, and plotting the cell's membrane potential.

5. Make sure to add these new files to your github repository and commit a new version of it, then push to the website before submitting to Gradescope so that your bonus work is also submitted.