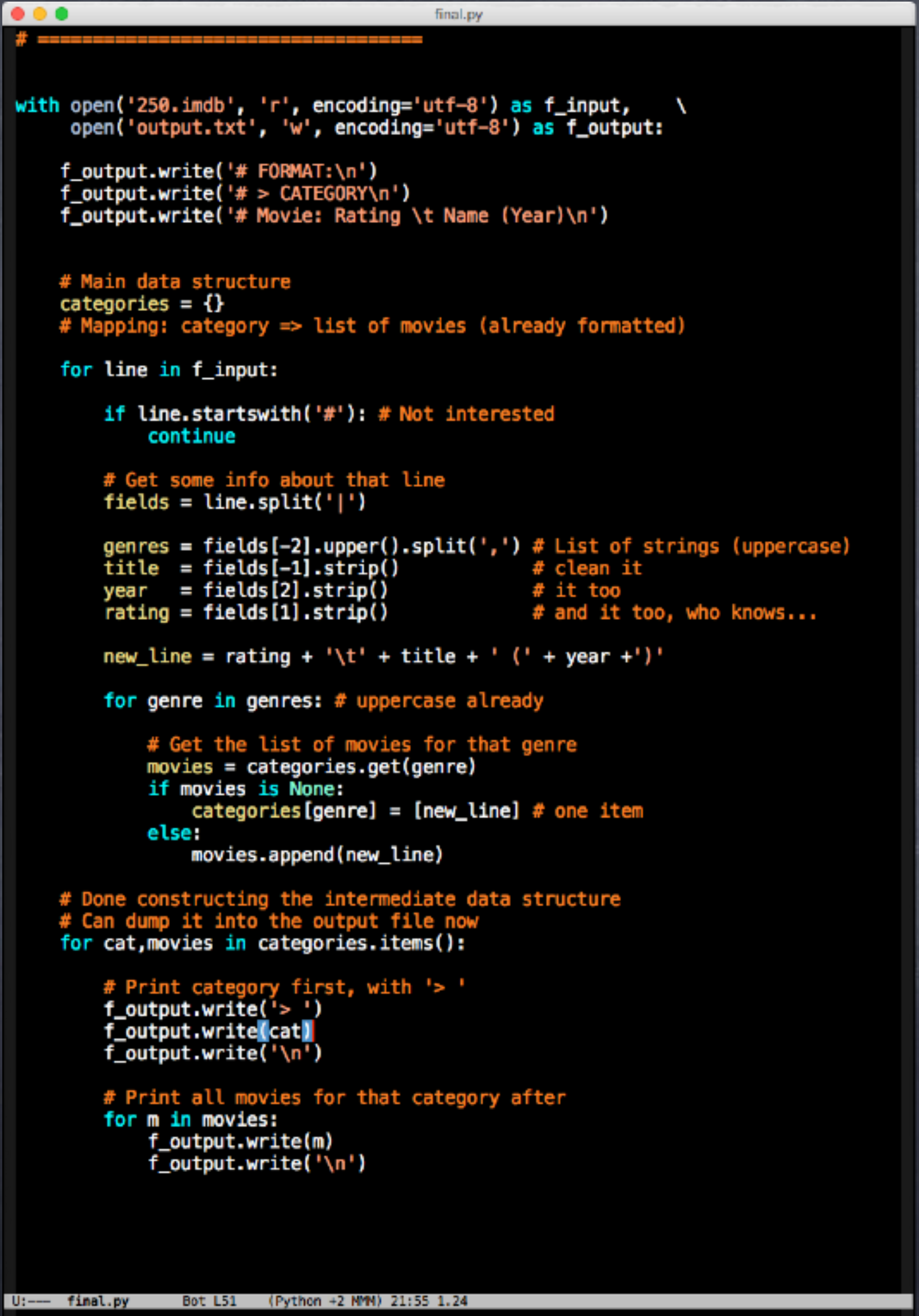
```
def functionName( parameters ):
    "Documentation" # Usually a triple quote

    someValue = ... # Make it something

    # Some code with someValue
    # using parameters (or not!)
    #
    # Some more code

    return someValue
```



```
# ===================================


with open('250.imdb', 'r', encoding='utf-8') as f_input,    \
     open('output.txt', 'w', encoding='utf-8') as f_output:

    f_output.write('# FORMAT:\n')
    f_output.write('# > CATEGORY\n')
    f_output.write('# Movie: Rating \t Name (Year)\n')


    # Main data structure
    categories = {}
    # Mapping: category => list of movies (already formatted)

    for line in f_input:

        if line.startswith('#'): # Not interested
            continue

        # Get some info about that line
        fields = line.split('|')

        genres = fields[-2].upper().split(',') # List of strings (uppercase)
        title  = fields[-1].strip()            # clean it
        year   = fields[2].strip()             # it too
        rating = fields[1].strip()             # and it too, who knows...

        new_line = rating + '\t' + title + ' (' + year +')'

        for genre in genres: # uppercase already

            # Get the list of movies for that genre
            movies = categories.get(genre)
            if movies is None:
                categories[genre] = [new_line] # one item
            else:
                movies.append(new_line)

    # Done constructing the intermediate data structure
    # Can dump it into the output file now
    for cat,movies in categories.items():

        # Print category first, with '> '
        f_output.write('> ')
        f_output.write(cat)
        f_output.write('\n')

        # Print all movies for that category after
        for m in movies:
            f_output.write(m)
            f_output.write('\n')
```