**Hottest Year**

from mrjob.job import MRJob

from mrjob.step import MRStep

class Max\_temp(MRJob):

def mapper(self, \_, line):

# Extract year and temperature from the line

year, temp = line.split(',')

yield year, float(temp)

def reducer\_find\_max\_temp(self, year, temps):

max\_temp = max(temps)

yield None, (year, max\_temp)

def reducer\_find\_year\_with\_max\_temp(self, \_, year\_temp\_pairs):

year\_with\_max\_temp = max(year\_temp\_pairs, key=lambda x: x[1])

yield year\_with\_max\_temp

def steps(self):

return [

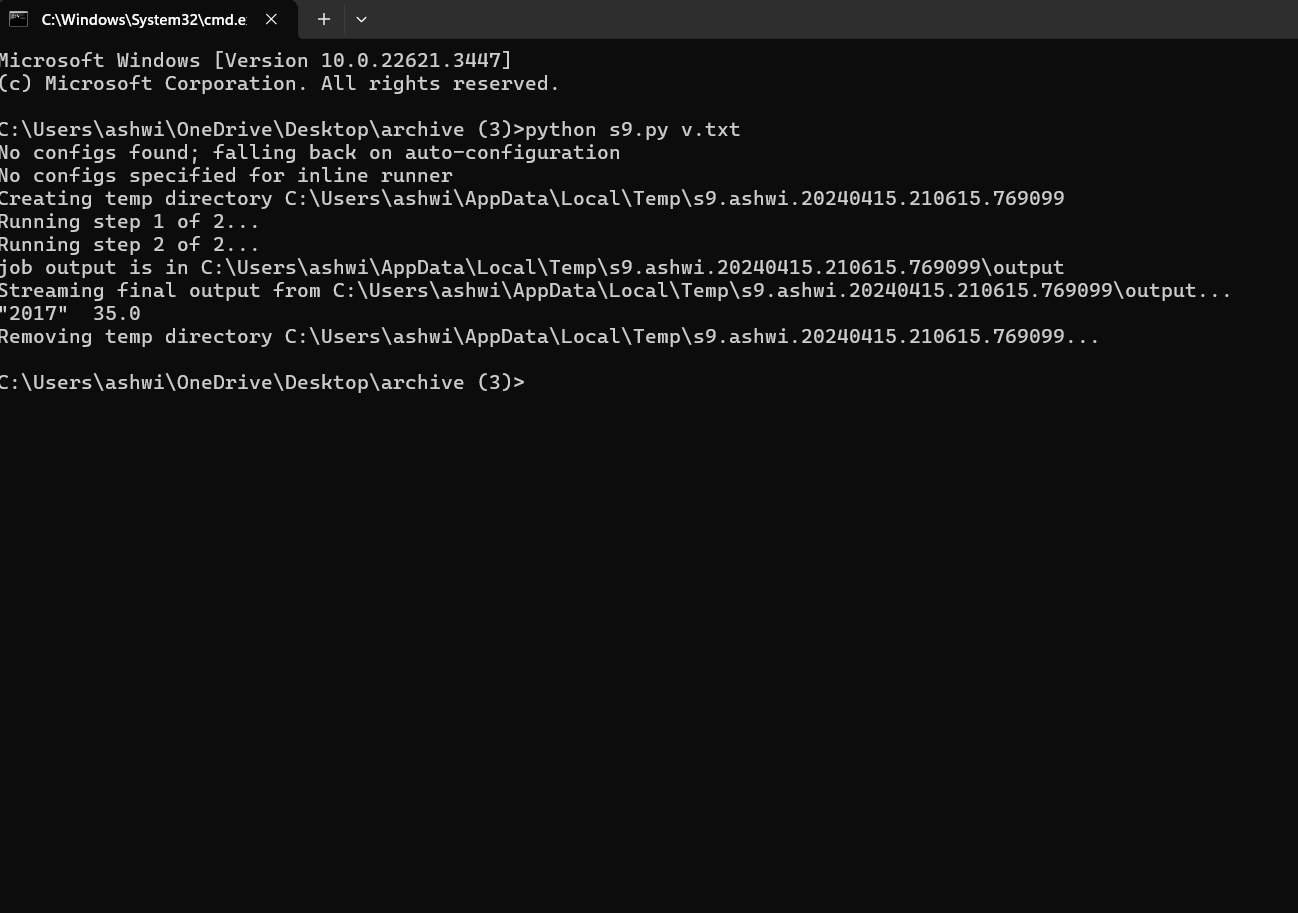
MRStep(mapper=self.mapper, reducer=self.reducer\_find\_max\_temp),

MRStep(reducer=self.reducer\_find\_year\_with\_max\_temp)

]

if \_\_name\_\_ == "\_\_main\_\_":

Max\_temp.run()



**Coolest Year**

from mrjob.job import MRJob

from mrjob.step import MRStep

class Min\_temp(MRJob):

def mapper(self, \_, line):

# Extract year and temperature from the line

year, temp = line.strip().split(',')

yield year, float(temp)

def reducer\_find\_min\_temp(self, year, temps):

min\_temp = min(temps)

yield None, (year, min\_temp)

def reducer\_find\_year\_with\_min\_temp(self, \_, year\_temp\_pairs):

year\_with\_min\_temp = min(year\_temp\_pairs, key=lambda x: x[1])

yield year\_with\_min\_temp

def steps(self):

return [

MRStep(mapper=self.mapper, reducer=self.reducer\_find\_min\_temp),

MRStep(reducer=self.reducer\_find\_year\_with\_min\_temp)

]

if \_\_name\_\_ == "\_\_main\_\_":

Min\_temp.run()

