EE466 Fall 2019 HW 2
Lewis Collum (0621539) EE/CE
Due: September 25, 2019

1.7.b

```python
class Compiler:
    def __init__(self, instructions, executionTime):
        self.instructions = instructions
        self.executionTime = executionTime

    @property
    def instructionRate(self):
        return self.instructions / self.executionTime

compilers = {
    'A': Compiler(
        instructions = 1.0e9,
        executionTime = 1.1),
    'B': Compiler(
        instructions = 1.2e9,
        executionTime = 1.5)
}

instructionRateRatio = compilers['A'].instructionRate / compilers['B'].instructionRate
print((f"Compiler A's processor has a clock that is {instructionRateRatio:.2}"
       f" faster than Compiler B's processor"))
```

```
Compiler A's processor has a clock that is 1.1 faster than Compiler B's processor
```

1.8.2

Processor Name	Clock Rate (GHz)	Voltage (V)	Static Power (W)	Dynamic Power (W)
Pentium 4 Prescott	3.6	1.25	10	90
Core i5 Ivy Bridge	3.4	0.9	30	40

```python
from collections import namedtuple
Processor = namedtuple('Processor', ['clockRate', 'voltage', 'staticPower', 'dynamicPower'])
processors = {row[0]: Processor(*row[1:]) for row in table}

for name, processor in processors.items():
    totalPower = processor.staticPower + processor.dynamicPower
    staticPowerPercentage = processor.staticPower / totalPower
    staticToDynamicRatio = processor.staticPower / processor.dynamicPower

    print((f"{name}:\n"
           f"  Static Power Percentage = {staticPowerPercentage*100:.3}%\n"
           f"  Static to Dynamic Ratio = {staticToDynamicRatio:.2}\n"))
```

```
Pentium 4 Prescott:
  Static Power Percentage = 10.0%
  Static to Dynamic Ratio = 0.11

Core i5 Ivy Bridge:
  Static Power Percentage = 42.9%
  Static to Dynamic Ratio = 0.75
```

1.9.1

Creating the Table

```python
from __future__ import annotations
from collections import namedtuple

clockRate = 2e9

ProgramInstructions = namedtuple('ProgramInstructions', ['arithmetic', 'loadStore', 'branch'])
instructionCount = ProgramInstructions(
    arithmetic = 2.56e9,
    loadStore = 1.28e9,
    branch = 256e6)
cyclesPerInstruction = ProgramInstructions(
    arithmetic = 1,
    loadStore = 12,
    branch = 5)

singleProcesssorExecutionTime = (cyclesPerInstruction.arithmetic*instructionCount.arithmetic/0.7 +
          cyclesPerInstruction.loadStore*instructionCount.loadStore/0.7 +
          cyclesPerInstruction.branch*instructionCount.branch)/clockRate

table = [["Processors", "Execution Time (s)", "Speed Up (s)"],
         [1, singleProcesssorExecutionTime, 0.0]]

for i in [2**x for x in range(1, 9)]:
    executionTime = (cyclesPerInstruction.arithmetic*instructionCount.arithmetic/(0.7*i) +
              cyclesPerInstruction.loadStore*instructionCount.loadStore/(0.7*i) +
              cyclesPerInstruction.branch*instructionCount.branch)/clockRate
    speedUp = singleProcesssorExecutionTime - executionTime
    table.append([i, executionTime, round(speedUp, 3)])

print(table)
```

Processors	Execution Time (s)	Speed Up (s)
1	13.44	0.0
2	7.04	6.4
4	3.84	9.6
8	2.24	11.2
16	1.44	12.0
32	1.04	12.4
64	0.84	12.6
128	0.74	12.7
256	0.69	12.75

Plotting the Table Results

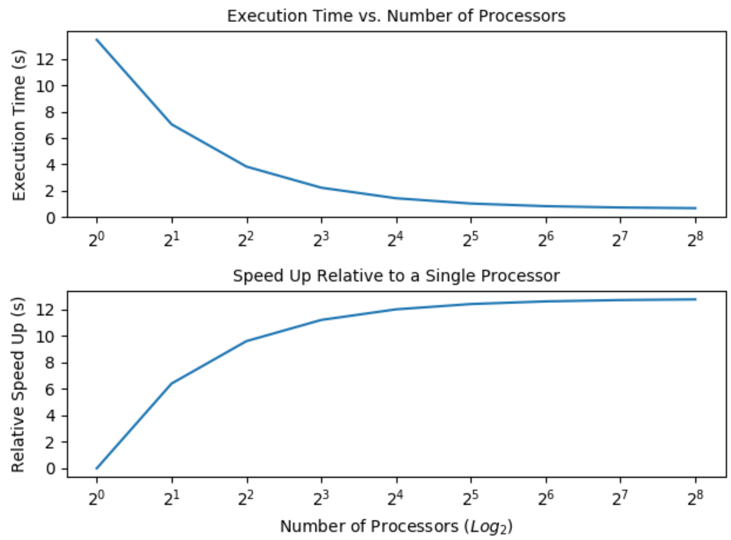
```python
import matplotlib.pyplot as pyplot
import numpy

table = numpy.asarray(table)
processors = table[1:, 0].astype(numpy.float)
executionTimes = table[1:, 1].astype(numpy.float)
speedUp = table[1:, 2].astype(numpy.float)

figure, axes = pyplot.subplots(2)
axes[0].set_title('Execution Time vs. Number of Processors', fontsize=10)
axes[0].set_ylabel('Execution Time (s)')
axes[0].plot(processors, executionTimes)
axes[0].set_xscale('log', basex=2)
axes[0].yaxis.set_ticks(numpy.arange(0, 14, 2))
```

```python
axes[1].set_title('Speed Up Relative to a Single Processor', fontsize=10)
axes[1].set_ylabel('Relative Speed Up (s)')
axes[1].plot(processors, speedUp)
axes[1].set_xlabel(r'Number of Processors ($Log_2$)')
axes[1].set_xscale('log', basex=2)
axes[1].yaxis.set_ticks(numpy.arange(0, 14, 2))
figure.tight_layout()

fileName = f"figure/{homework}_{problem}.png"
figure.savefig(fileName)
return fileName
```



1.14.1

This one is not from the textbook