Mezbah Uddin
Ryerson ID:500793378
Assignment 2, #8001

Question 1

Amdahl's law = T(N) = B + (T(1) - B) / N
Given the execution times of each block, the non-parallelizable fraction is 120m/190m= .6315 fraction of the total execution time.
Therefore, $S_{\text{latency}}[20]$ = $\frac{1}{\left(1-\frac{120}{190}\right)+\frac{120/190}{20}}$ = 2.50

Question 2

As s tends to infinity, the limit of $S_{\text{latency}}[20]$ is ${}^{1}\!/_{1-\frac{120}{190}}$ = 2.7777

Question 3

Karp-flatt metric = $\frac{\frac{1}{\frac{190}{127}}-\frac{1}{10}}{1-\frac{1}{10}}$ = .63

Question 4

Since there are 10 processors and the for loop goes for i = 1 to 10, probably easiest way to parallelize is by assigning each processor to each i. In that way, each processor will share a load of 1 i and 1/10 of share of m for loop.

Therefore each p has to do the following –

```
for i = 1
        quickly_do_some_prep_work(1)
                for j = 1 to 100: compute_something_expensive_and_non_parallelizable(i,
        j)
```

Question 5

Assuming time needed for the non-parallelizable algorithm follows a uniform random distribution with a mean of $\frac{1}{2}[1 + 72000] = 36000.5\, s = 10\ hours$
In the above case , the following algorithm can be constructed to map the task-

```
x= ran[1,72000]

If x<10:
        For p in range[1,10]:
           quickly_do_some_prep_work(1)
                for j = 1 to 100:
                        compute_something_expensive_and_non_parallelizable(i, j)
if x>10:
        for p in range[1,5]

                quickly_do_some_prep_work(1)
                for j = 1 to 100:
                        compute_something_expensive_and_non_parallelizable(i, j)
```

To efficiently parallelize, one possible solution could be **master-slave parallel computing scheduling** where each processor doesn't sit idle at any time. Each job consists of a preprocessing task, a slave task and a post-processing task that must be executed in this order. The pre- and post-processing tasks are to be processed by a master processor while the slave task is processed by a slave processor.