**Problem 7**

**a)**

X = 10 (1/sec)

S = 50ms

ρ = 10\*50\*10-3 = .5

U = ρ = .5

Average number jobs = .5/.5 = 1

Average response time = s/(1-U) = 50/.5 msec = 100msec

Average time spent in computer = 1(100) = 100msec

Max arrival rate → 50(10)-3[x] = 1

x = 103 / 50 = 20 jobs/second

**b)**

vs = σs­/s = (60-40)/(3.5(100)) = .11547

U = SX = 50[10-3] \* 10 → .5

S = 50ms

R = S/(2 – U)[2 – U(1 + vs2)/2]

= 50/(2 - .5)[2 – .5(1 + vs2)/2]

= 50 ms

Average number jobs = .5/.5 = 1

Average time spent in computer = 1(50) = 50ms

Max arrival rate = 20 jobs/second

**Problem 4**



#include <iostream>  
#include <cmath>  
#include <stdlib.h>  
**static const int** CYLINDERS = 900;  
**static const int** FILE\_CYLINDERS = 300;  
  
**double** t( **int** x) { **return** pow(x,.5); }  
  
**static int** getHeadStart() { **return** (**int**) ( (**double**)rand()/(**double**)RAND\_MAX \* 900.0 ); } //head location  
**static int** getDataStart() { **return** (**int**) ( (**double**)rand()/(**double**)RAND\_MAX \* 600.0); } //data location  
**static double** getRotationalDelay() { **return** ( (**double**)rand()/(**double**)RAND\_MAX \* 8.3333333333);} //rotational delay  
  
**static double** getAccessTime()  
{  
 **static int** headStart = getHeadStart();  
 **static int** dataStart = getDataStart();  
 **static double** rotationalDelay = getRotationalDelay();  
 **return** ( t( abs (headStart-dataStart) ) + rotationalDelay );  
}  
**static void** solve2()  
{  
 **static double** sumTime = 0;  
 **for** ( **int** i = 0; i < 100; i++) { sumTime = sumTime + getAccessTime(); }  
 std::cout << "Average access time: " << sumTime/100.0;  
}  
**int** main()  
{  
 solve2();  
 **return** 0;  
}