**Universidade Federal de Uberlândia**

**GBC065 – Modelagem e Simulação**

**TRABALHO 3**

*Alunos:* -João Vitor Rodrigues de Alcântara Ferreira

-Thiago Lucas Carvalho

Exercícios:

**Exercise 3.1.2**

(a) Relative to the steady-state statistics in Example 3.1.3 and the statistical equations in Section 1.2, list all of the consistency checks that should be applicable.



**w é a média de jobs em espera**

**d é a média de jobs em delay**

**s é a média de jobs em serviço**

**l é o tempo médio no nó**

**x é o tempo médio de ocupação no server em %**

**q é o tempo médio na fila**

**Elas se relacionam da seguinte maneira: w = d + s**

**Para verificarmos o tempo médio no nó temos: l = q + x**

(b) Verify that all of these consistency checks are valid.

**Para verificar se a média de jobs em fila é válido temos:**

**3.83 = 2.33 + 1.50**

**3.83 = 3.83**

**Essa consistência é válida**

**Para verificar se o tempo médio no nodo é válido temos:**

**1.92 = 1.17 + 0.75**

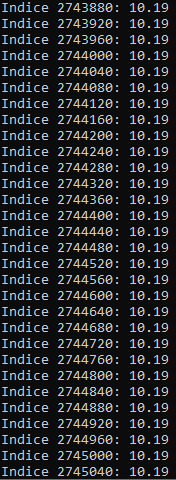
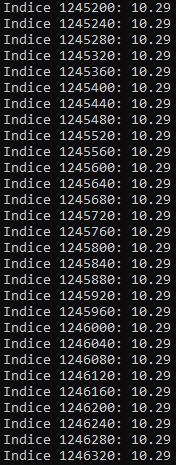
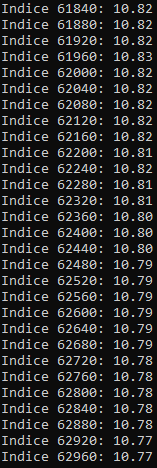
**1.92 = 1.92**

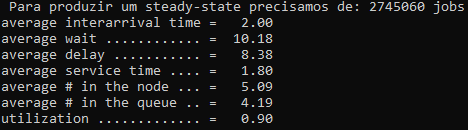
**Essa consistência é válida**

**Exercise 3.1.4**

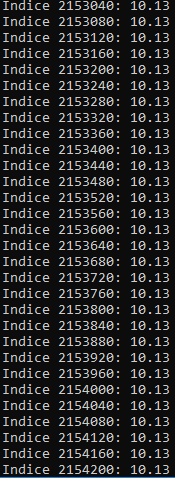
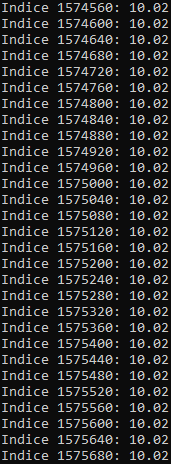
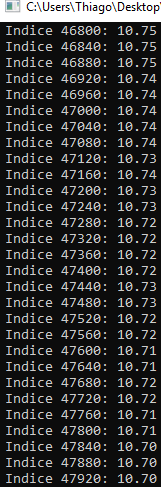
(a) Conduct a transition-to-steady-state study like that in Example 3.1.3 except for a service time model that is Uniform(1.3, 2.3). Be specific about the number of jobs that seem to be required to produce steady-state statistics.

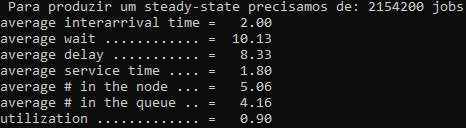
**Seed 2121212**

****

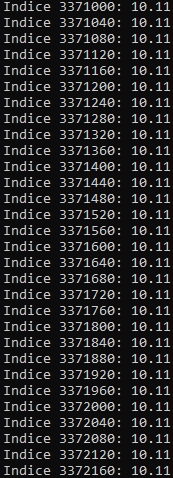
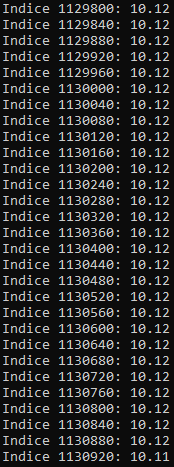
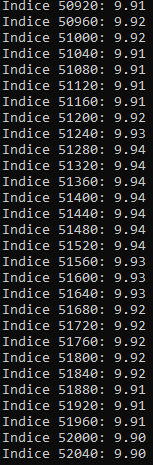
****

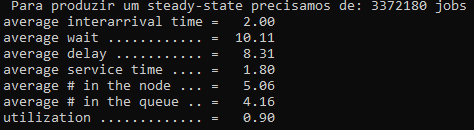
**Seed 54321**

****

****

**Seed 12345**

****

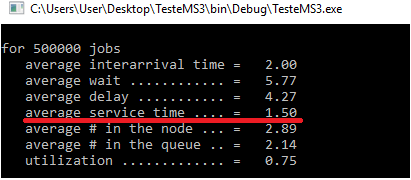
****

(b) Comment.

**O steady-state varia constantemente até que os valores vão convergindo até chegar em um estado quase constante.**

**Exercise 3.1.5**

(a) Verify that the mean service time in Example 3.1.4 is 1.5.

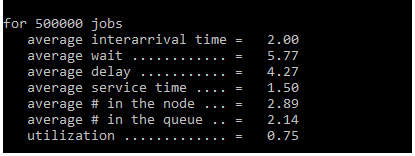


**O exemplo 3.1.4 mostra o seguinte service time :**

****

**Apesar deste resultado ser igual, caso o número de jobs fosse menor (ex: 10000 padrão), o service time seria 1.49.**

(b) Verify that the steady-state statistics in Example 3.1.4 seem to be correct.



**Os resultados do exemplo 3.1.4 são obtidos mais precisamente de acordo com que o número de Jobs definido é aumentado, chegando a um valor exato ao do livro com 500.000 mil jobs processados.**

(c) Note that the arrival rate, service rate, and utilization are the same as those in Example 3.1.3, yet all the other statistics are larger than those in Example 3.1.3. Explain (or conjecture) why this is so. Be specific.

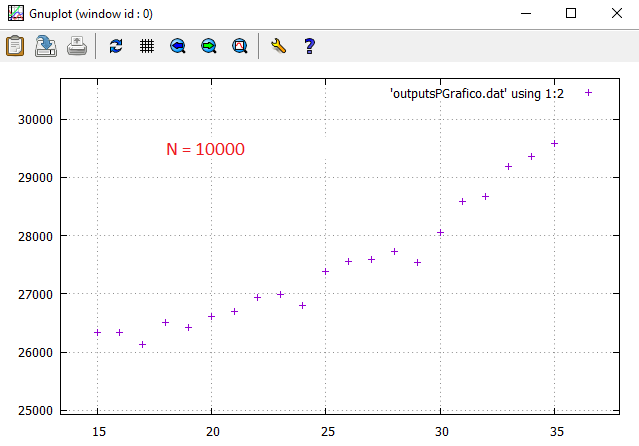
**As taxas no exemplo 3.1.4 serem maiores do que as do ex. 3.1.3 são explicadas pela sensibilidade das medidas de desempenho à distribuição do tempo de serviço, destacando também a importância de usar-se um modelo de tempo de serviço o mais preciso possível.**

**Exercise 3.1.6**

(a) Modify program sis2 to compute data like that in Example 3.1.7. Use the functions PutSeed and GetSeed from the library rng in such a way that one initial seed is supplied by the system clock, printed as part of the program’s output and used automatically to generate the same demand sequence for all values of s.

**Código em anexo**

(b) For s = 15, 16, . . . , 35 create a figure (or table) similar to the one in Example 3.1.7.



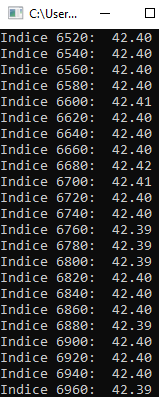
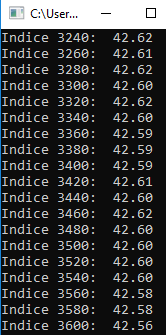
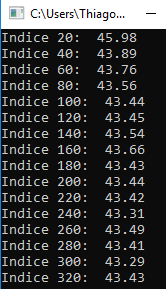
(c) Comment.

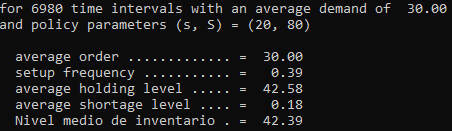
**Pelo exemplo observado no livro e pelos testes feitos, pode-se observar que o uso de seed’s definidas inicialmente para seed’s geradas por alguma política randômica é que a variabilidade das medidas e resultados mudam, sendo a randômica geradora de resultados mais naturais e satisfatórios.**

**Exercise 3.1.7**

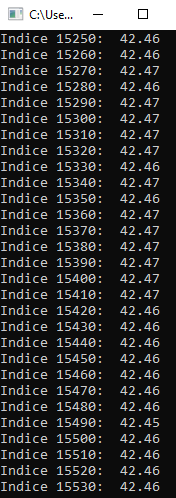
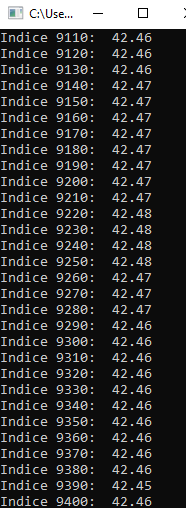
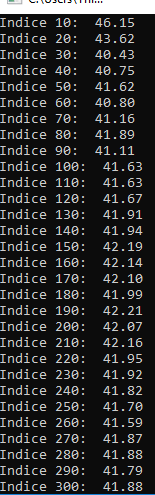
(a) Relative to Example 3.1.5, if instead the random variate sequence of demands are generated as d i = Equilikely(5, 25) + Equilikely(5, 25) i = 1, 2, 3, . . . then, when compared with those in Example 3.1.6, demonstrate that some of the steady-state statistics will be the same and others will not.

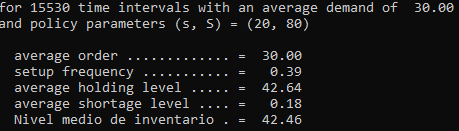
**Seed 2121212**

****

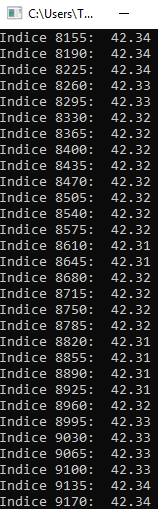
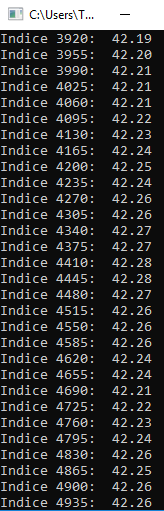
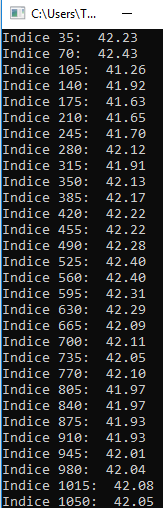


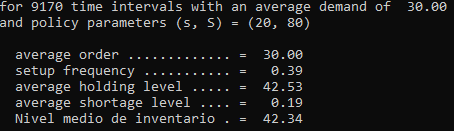
**Seed 54321**

****

****

**Seed 12345**

****

****

(b) Explain why this is so.

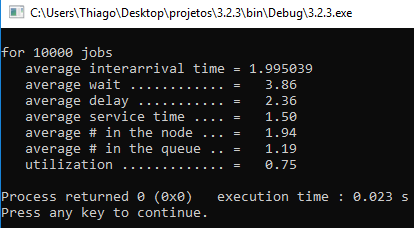
**O que acontece é que os valores de Equilikely tem um intervalo menor do que o do exemplo, e quando executamos o programa ele retorna valores menores para o holding level e shortage level.**

**Exercise 3.2.3** Modify program ssq2 as suggested in Example 3.2.7 to create two programs that differ only in the function GetService. For one of these programs, use the function as implemented in Example 3.2.7; for the other program, use

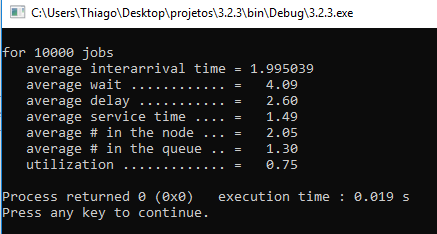
double GetService(void) { SelectStream(2); /\* this line is new \*/ return (Uniform(0.0, 1.5) + Uniform(0.0, 1.5)); }

(a) For both programs verify that exactly the same average interarrival time is produced (print the average with d.dddddd precision). Note that the average service time is approximately the same in both cases, as is the utilization, yet the service nodes statistics w̄, d, ̄l, and q̄ are different.

Código 1:



Código 2:



(b) Why?

**A média de tempo de serviço fica parecido nos dois códigos pois a função é a função uniform Uniform(0.0,1.5) + Uniform(0.0,1.5) é quase a mesma coisa que Uniform(1.0,2.0)**

**Exercise 3.3.10** Modifique o programa sis2 para incluir lag na entrega do fornecedor e construir gráficos similares ao Example 3.3.4. Para reproduzir o gráfico sem lag faça com que o Delta (variável que implementa o lag) seja igual a zero e compare os resultados.