Simulação e Computação Científica

**Relatório**

**Estudo do funcionamento dum sector duma fábrica**

Conteúdo

**Não foi encontrada nenhuma entrada de índice.**

**Introdução**

Este projeto foi desenvolvido em Python e trata-se de uma simulação de um sector de uma fábrica.

Neste relatório vamos explicar como validamos o nosso simulador, como é pedido na alínea b), vamos analisar o funcionamento do simulador, detetar as falhas deste e discutir as possíveis situações de melhoramento referidas na alínea c) – “Aumentar o número de máquinas nas secções críticas” ou “substituí-las por 2 máquinas mais rápidas (média de processamento de 1.7 minutos), mas mais caras do que a solução anterior.”.

**Arquitetura do simulador**

**CLASSES**

*Simulador*

Esta é a classe principal do nosso trabalho. Gere os eventos que vão acontecer assim como os serviços. Reúne as estatísticas de todos os serviços e escreve-as para um ficheiro.

Os seus atributos são:

* **nome** – nome do simulador
* **numero\_pecas** - número de peças diferentes que passam pelas secções
* **media\_chegada\_pecas** – é uma lista que guarda a média das distribuições de chegada das peças
* **tempo** – inteiro que funciona como relógio do simulador, sendo sempre inicializado a 0
* **horas** – guarda as horas de produção por dia
* **dias** – guarda o número de dias da simulação
* **matriz\_serviços** – corresponde a uma lista de listas em que contém todas as peças que passaram naquele sector e a cada uma dessas peças está associada uma lista de serviços que a afetam
* **lista** – é lista de eventos que guarda todos os eventos da simulação, existe apenas uma por simulador
* **tipo\_pecas** – é uma lista com os diferentes tipos de peças
* **divida** – inteiro que guarda a dívida da fábrica (variável usada apenas na alínea e) )
* **pausa** – boolean que permite percorrer o simulador dia-a-dia caso pretendermos analisar as estatísticas ao fim de cada dia e não no fim dos dias todos de simulação
* **debug** – boolean que permite decidir se queremos imprimir a sequência de eventos que decorreram
* **registrar** – boolean que permite escolher se pretendemos guardar as estatísticas num ficheiro

No início é criada uma instância do simulador e este é inicializado, logo de seguida é chamada a função executa.

A função executa começa por criar o evento de chegada para cada peça, começa por zerar o relógio e enquanto não forem completadas as horas de trabalho o simulador retira um o primeiro evento da lista, atualiza o relógio para o instante desse evento, atualiza as estatísticas e executa evento. Este processo repete-se até que o relógio do simulador atinja as horas de trabalho totais do dia.

Seguidamente são impressas as estatísticas para aquele dia e o número de dias que foram executados aumenta. No fim é sempre verificado se ainda há dias por executar.

Assim que o número total de dias a executar é atingido, o simulador termina.

*Serviço*

Esta classe representa uma secção com a sua fila de espera associada e com um número de máquinas específico para servir as peças.

É aqui que são feitos todos os cálculos estatísticos e armazenados todos os valores necessários para esses mesmos cálculos.

Os principais métodos são o: inserePeca (coloca a peça na máquina(s) do serviço ou na fila de espera caso a máquina(s) esteja ocupada) e retiraPeca (retira a peça do serviço e caso haja alguma peça na fila de espera encaminha-a para a máquina livre).

*Evento*

É desta classe que são derivados todos os eventos. Nos seus atributos encontra-se o simulador, o serviço, a peça e o instante em que o evento ocorre.

As duas funções definidas nesta classe são a função Chegada() que representa a chegada de uma peça a um serviço e a função Saida() que representa a saída da peça do serviço.

*Lista*

Esta classe contém os eventos do simulador guardados numa lista que está constantemente atualizada. As funções aqui incluídas são insere\_evento() que ao inserir o evento reordena a lista de modo a que no início da lista estejam sempre os eventos que vão ocorrer primeiro na simulação, a função retira\_evento() que retira sempre o primeiro evento da lista e por fim uma função que nos imprime a lista.

*Peça*

Esta classe representa a peça e apresenta como atributos: o tipo (inteiro que representa cada uma das peças que entra no sector), o nome (se se trata da peça A ou peça B) e o custo da peça.

*Registrador*

Esta classe permite-nos escrever as estatísticas para um ficheiro.

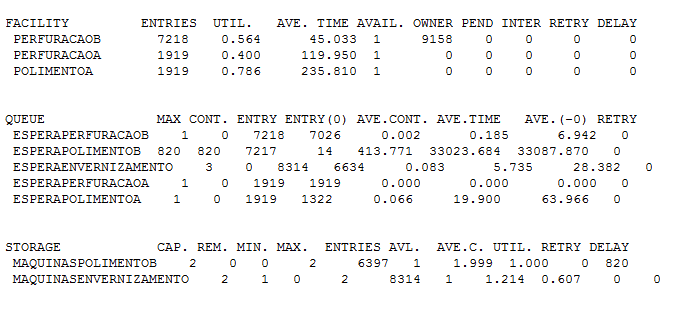
Encontra a pasta onde o programa está a correr e cria uma pasta nesse diretório. Temos a função comeca\_registo(nome) que recebe o nome que quisermos dar ao ficheiro, caso não seja dado nome ele atribui a data e hora com que foi criado o ficheiro como nome, cria-o e escreve um texto de introdução. A função regista() permite-nos adicionar informação ao ficheiro. Por fim temos uma função que serve apenas para fechar o ficheiro.

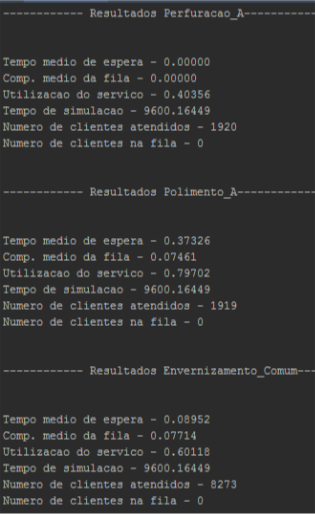
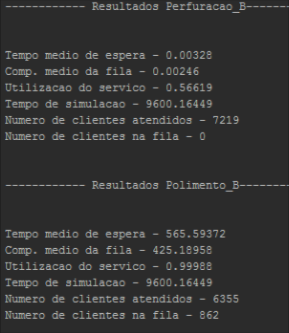
**Validação do Simulador**

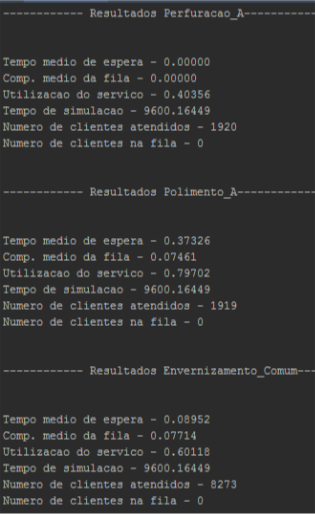
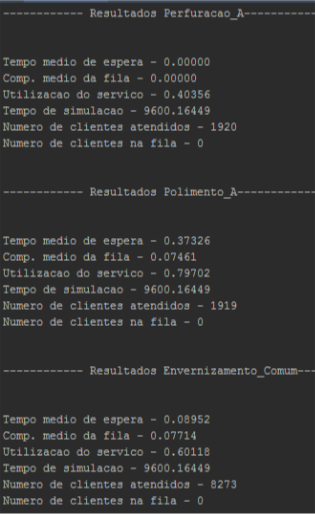
*Validação externa*

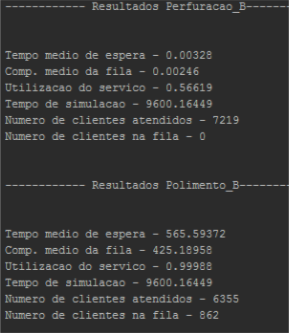
Para esta validação, utilizamos o GPSS e usando os mesmos valores tanto no nosso simulador como no GPSS comparamos os resultados. Repetimos este processo várias vezes. Ao comparar temos resultados muito semelhantes de ambos os lados, o que nos permite concluir que o nosso simulador está a fazer cálculos corretos.

Abaixo vamos incluir as imagens que nos permitem fazer a comparação.









Há apenas uma ligeira diferença no número de entrada das peças no envernizamento comum, isto deve-se ao facto de mais peças ficarem retidas na fila de espera do polimento B.

*Validação interna*

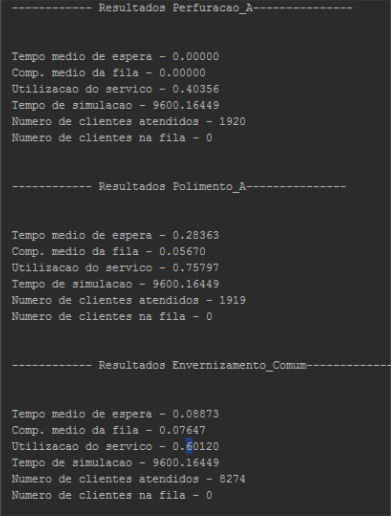
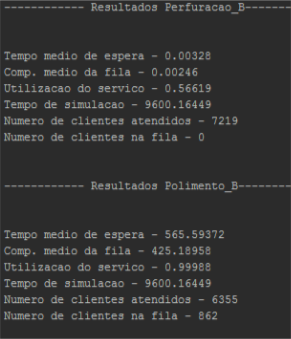
**Estabilidade**

*Validação por análise de parâmetros*

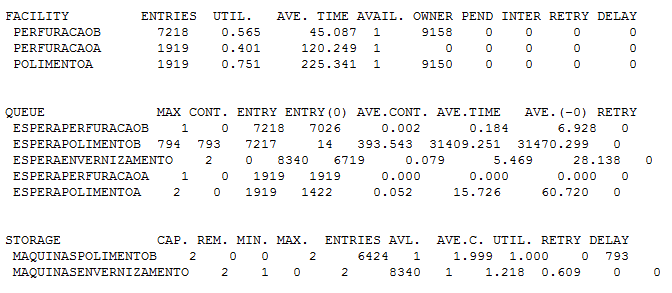
Vamos agora fazer variar os diferentes parâmetros (médias, tempo de chegada, tempo de paragem e número de máquinas), aumentando e diminuindo 5% em relação aos seus valores originais e vamos verificar os efeitos destas alterações no nosso sistema, para ter a certeza que coincidem com os valores teóricos esperados.

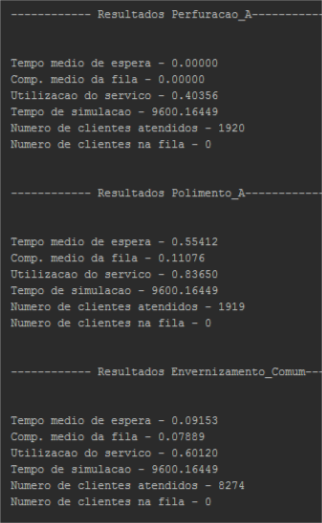
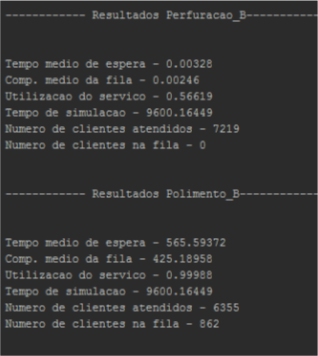
Estamos a validar também cada uma das alterações feitas com o GPSS.

**Variação da média do Polimento A**

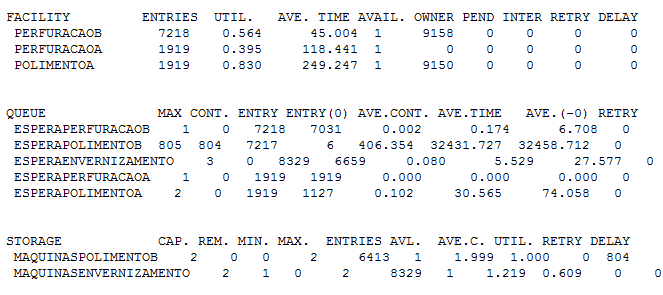


Média do polimento A = 3.8

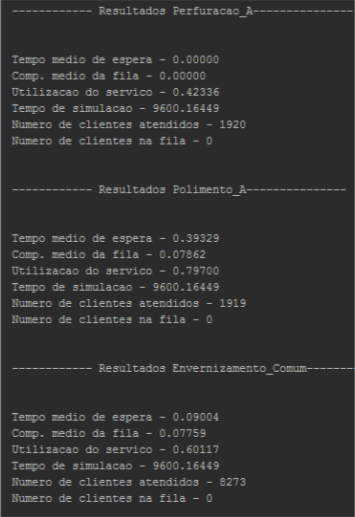
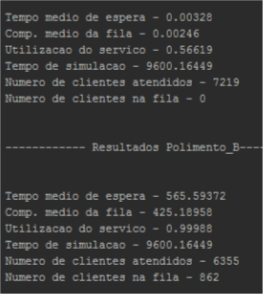




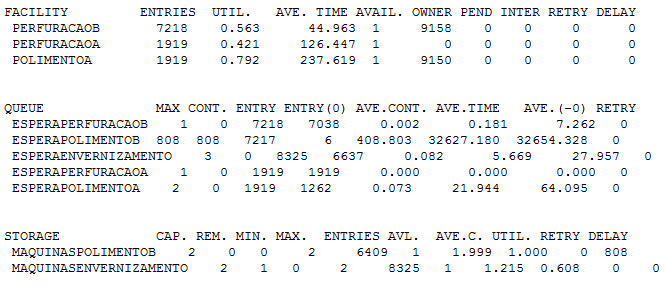
Média do polimento A = 4.2

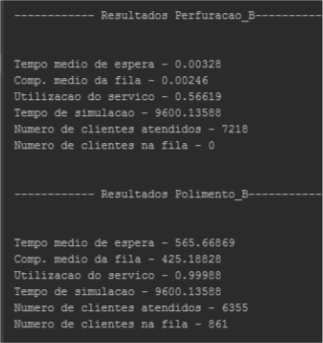
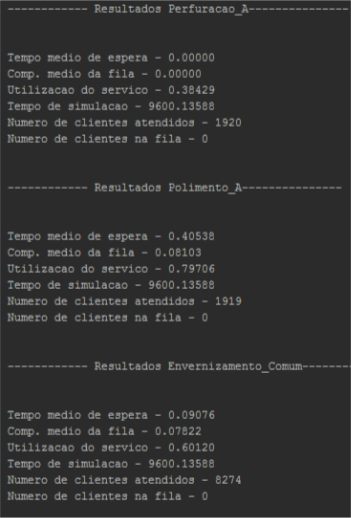


**Variação da média da Perfuração A**

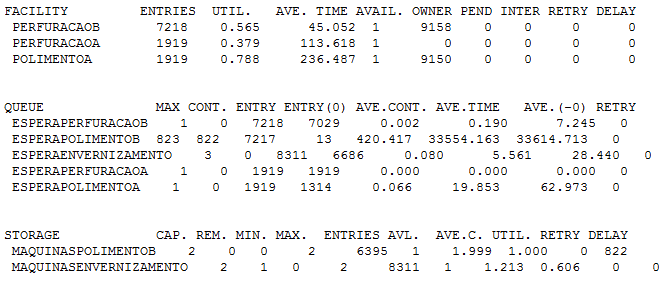


Média da Perfuração A = 2.1

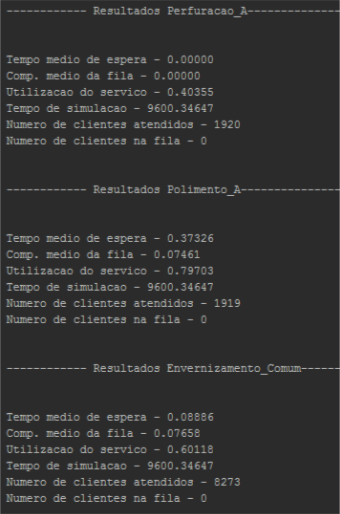
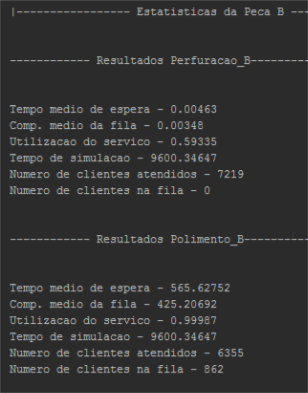




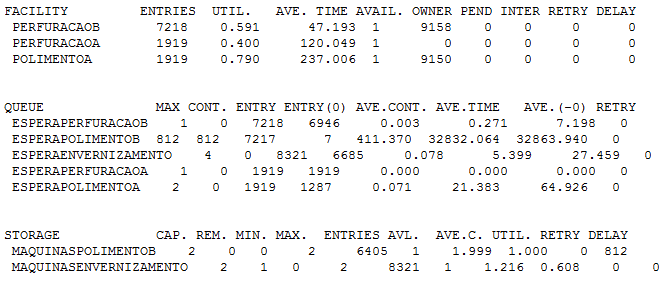
Média da Perfuração A = 1.9

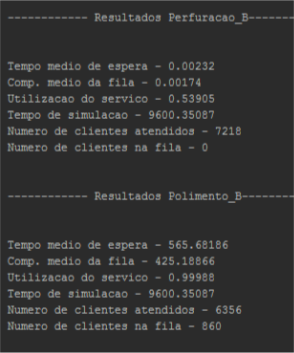
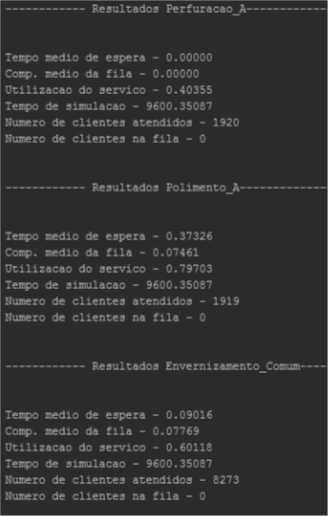


**Variação da média da Perfuração B**

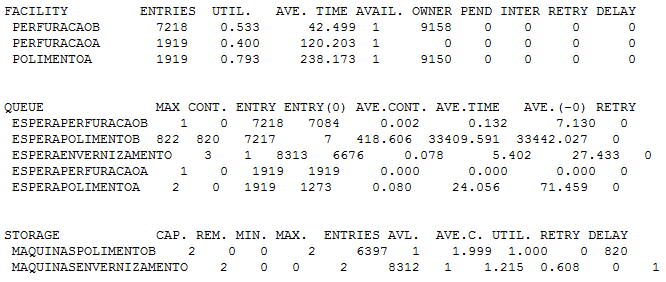


Média da Perfuração B = 0.7875

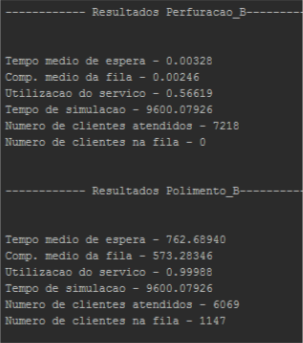
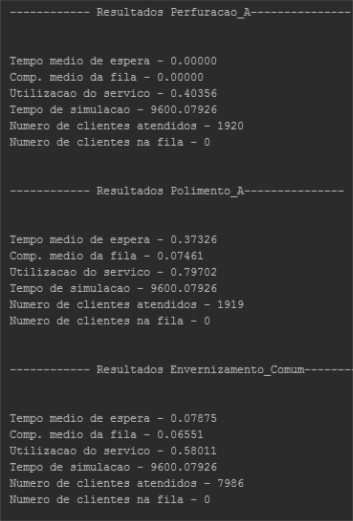




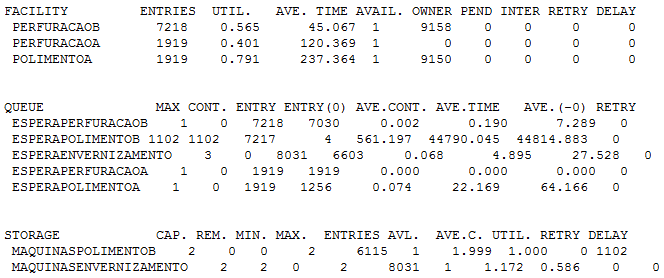
Média da Perfuração B = 0.7125

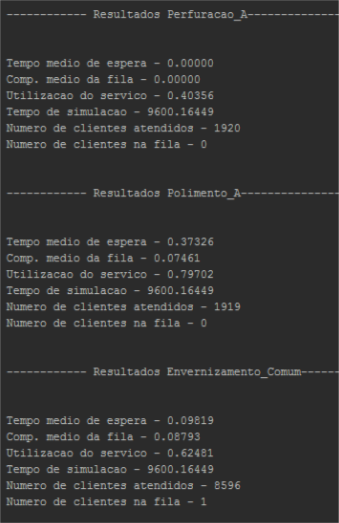
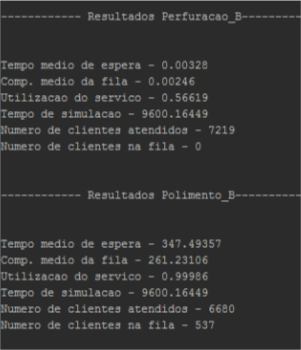


**Variação da média da Polimento B**

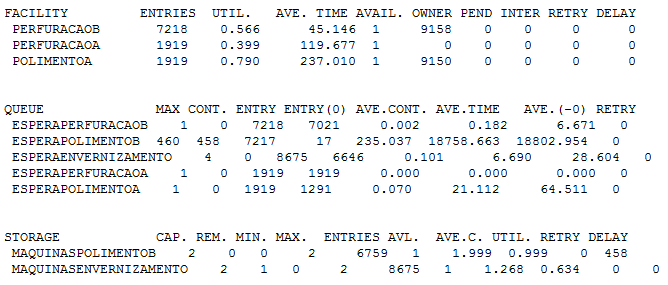


Média do Polimento B = 3.15

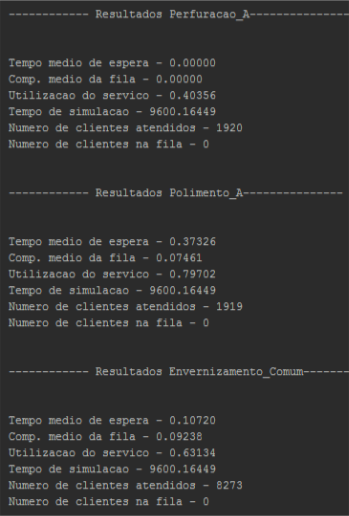
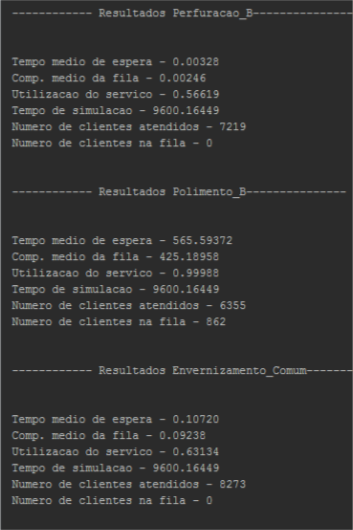




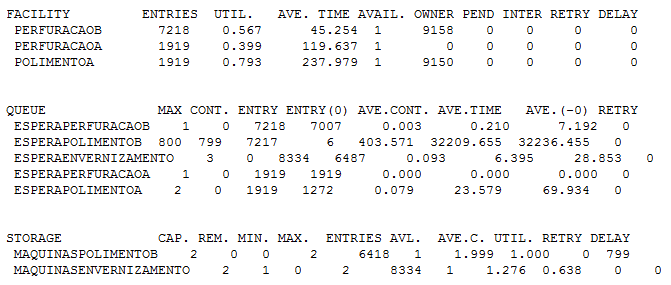
Média do Polimento B = 2.85

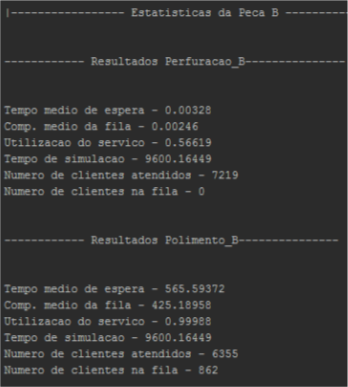
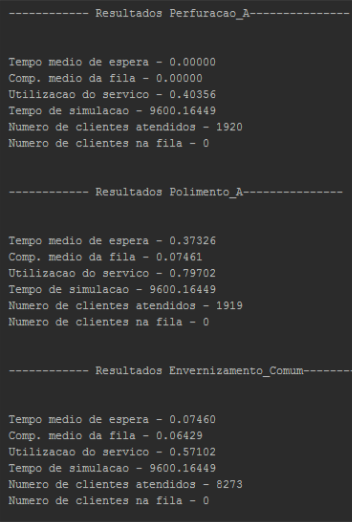


**Variação da média de Envernizamento**

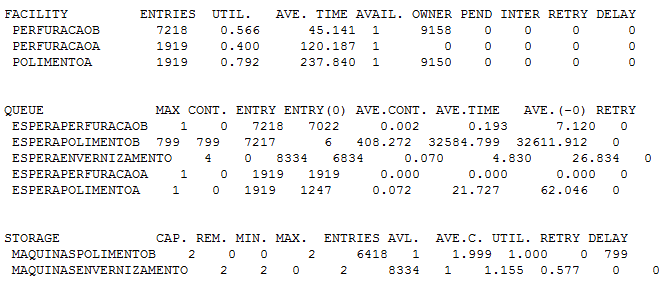


Média de Envernizamento = 1.47

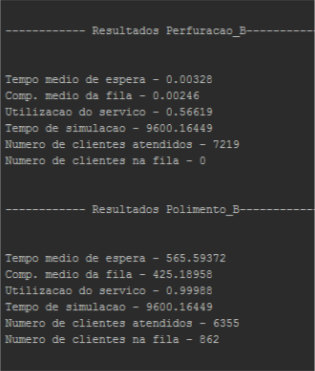
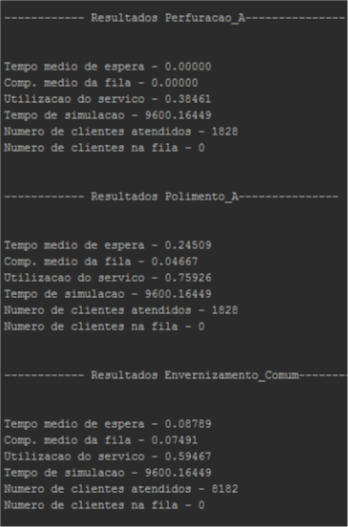




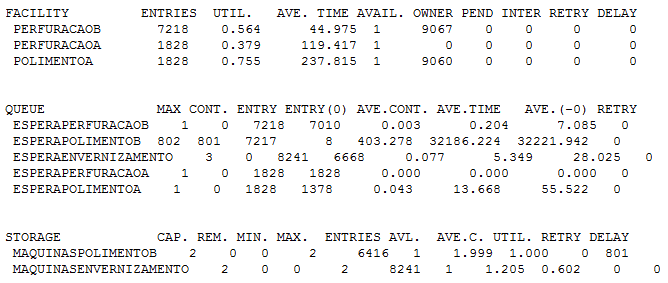
Média de Envernizamento = 1.33

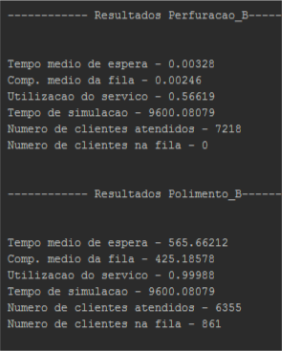
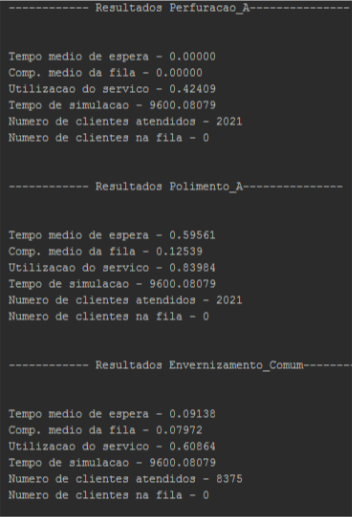


**Variação da** **média do tempo de chegada das peças A**



média do tempo de chegada das peças A = 5.25





média do tempo de chegada das peças A = 4.75

