Segundo Trabalho de Programação M&S

Esse trabalho foi desenvolvido pelos alunos Breno Corrêa e Salomão Alves. Todo o desenvolvimento foi utilizando a linguagem de programação Python, onde montamos um programa que simula um sistema de atendimento com ou sem fila.

Documentação

1. Introdução

Programa de simulação de sistema de atendimento com ou sem fila, totalmente desenvolvido utilizando python como linguagem de programação.

Seu funcionamento consiste em informações de entrada dadas pelo usuário, e com base nesses dados, simula o funcionamento desse sistema e gerar resultados relevantes para o usuário sobre essa execução.

2. Dados de entrada

Os dados de entrada são informações de configuração que serão utilizadas na simulação, como por exemplo, opções de fila, número de eventos que deseja simular, se o tempo de chegada e o tempo de serviço serão determinísticos ou aleatórios.

Em caso de escolha de valores aleatórios, será necessário o usuário passar o valor do lambda, pois é usada a distribuição exponencial para gerar um número aleatório. Porém, em caso de escolha de valores determinísticos, durante a execução da simulação será pedido valores para o usuário determinar.

3. Execução

Durante a execução da simulação, os eventos serão mostrados na tela, possibilitando ao usuário acompanhar as chegadas e saídas de clientes, a ocupação da fila, e o tempo de cada evento.

Alguns valores serão mostrados em siglas para facilitar a interface, os significados de cada uma são:

HC	Hora da Chegada
HS	Hora das Saídas
TR	Tempo da Simulação
ES	Estados dos Servidores
TF	Tamanho da Fila
TS	Tempo de Serviço
TEC	Tempo entre Chegada

4. Relatório

Ao final da simulação, será demonstrado alguns resultados importantes para o usuário, são eles:

- Tempo total de espera na fila
- Número total de clientes
- Número de clientes que esperaram
- Tempo livre do operador
- Tempo total de serviço
- Tempo total no sistema
- Tempo médio de espera da fila
- Probabilidade de um cliente esperar na fila
- Probabilidade do operador livre
- Tempo médio de serviço
- Tempo médio despendido no sistemas

5. Código

Primeiro será executado a *main(),* onde pegará os dados e criará três objetos; dois da classe *Operador()*, esses dois objetos serão utilizados para salvar as informações de cada operador durante a execução. O outro será da classe *Fila()*, a qual irá armazenar as informações dos clientes que entrarem na fila.

Caso for escolhido gerar os valores de TEC ou TS aleatoriamente, será executado a função *generate_var()*, que é responsável por gerar um valor aleatório utilizando a Transformação Inversa da distribuição exponencial.

Depois é chamado a função *simu()*, que será responsável por sincronizar as chegadas e saídas de cliente, dentro de um *for* que repete igual ao número de eventos que o usuário escolheu simular.

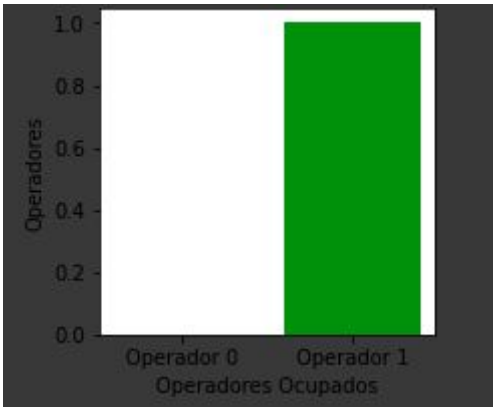
Para gerenciar para qual operador o serviço irá, mudamos a variável *state* e *hs* para vetores com dois elementos, onde o primeiro elemento representa o operador 0 e o segundo elemento o operador 1. Assim, tanto na função *chegada()*, quanto na função *saida()*, haverá alguns *if* que irão ajudar nesse gerenciamento: na função *chegada(),* se o serviço será iniciado no operador 0 ou 1 ou então irá para a fila; na função *saida()*, se o operador será liberado e já ocupado (ou seja, há serviço na fila) ou se será liberado e ficará livre, em ambos os casos, há um *if* interno que decidirá se a ação será feita no operador 0 ou 1.

Para cada evento de chegada ou saída, é gerado de acordo com a opção que o usuário escolheu (aleatório ou determinístico). Se o valor for determinístico, requisitará o usuário para digitar o valor de sua preferência, em caso de aleatório, chamará a função *generate_var().*

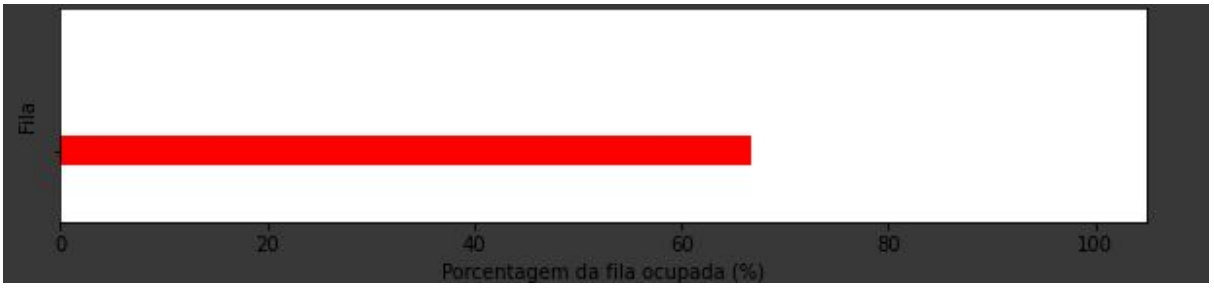
Ao final é chamado a função *printStats()* que pega os valores dos objetos *op0* e *op1* que são uma instância da classe *Operador()* (citado anteriormente), com esses valores é calculado os resultados finais da simulação e imprimidos na tela.

6. Gráficos

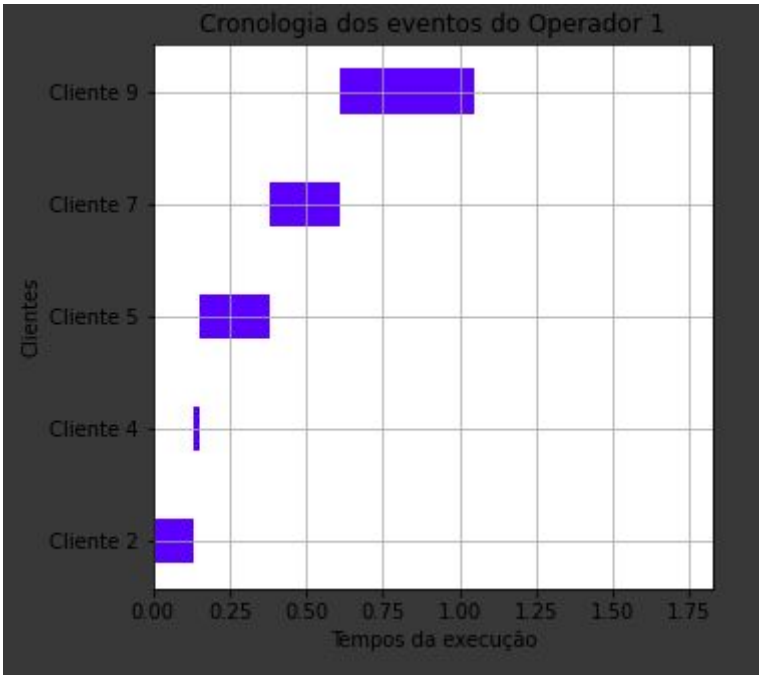
A função *grafico_ops()* (figura abaixo) irá plotar um gráfico com duas barras, a primeira representa o Operador 0 e a outra barra o Operador 1, quando a barra estiver completa em verde é porque aquele operador está sendo usado.



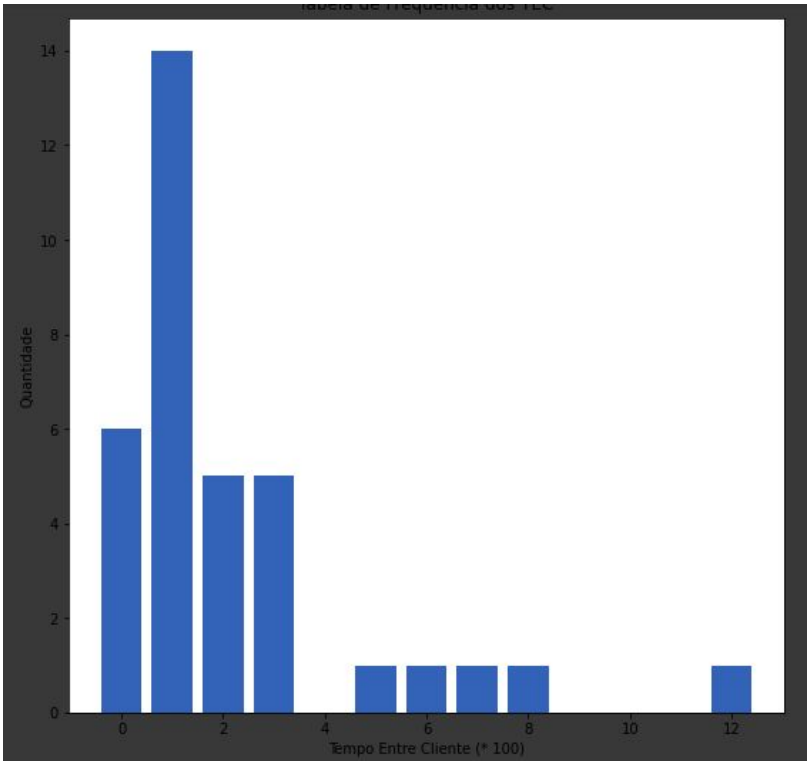
A função *grafico_fila()* (figura abaixo) irá plotar a porcentagem de ocupação da fila de espera, esse gráfico só irá aparecer quando definimos um tamanho fixo para fila.



A função *events_op0()* (figura abaixo) irá plotar a ocorrência durante o tempo dos eventos que usaram o Operador 0. A *events_op1()* faz a mesma coisa, só que para os eventos do Operador 1.



A *tec_seq()* e *ts_seq()* (figuras abaixo) irão plotar a tabela de frequência para o tempo entre cliente (tec) e tempo de serviço (ts), respectivamente.



As duas primeiras funções (*grafico_ops()*, *grafico_fila()*) serão plotadas a cada iteração do *for*, porém, o colab não plota logo após o *print*, por isso elas serão plotadas logo após o término do programa.

Já as outras, serão plotadas apenas no final da execução, como parte do relatório final.

7. Bibliotecas

numpy - Para utilizar arrays com mais fácil a manipulação de dados e para gerar um número uniformemente distribuído.

pandas - Para utilizar series e gerar a tabela de frequência

time - Para utilizar a função sleep()

matplotlib - Para plotar os gráficos