

Universidade da Beira Interior

Departamento de Informática



**Departamento de
Informática**

Computação Científica

**Implementação de um modelo de simulação de um serviço de atendimento
de uma repartição de Finanças**

Elaborado por:

M10304 - Cristiano Patrício

M10139 - Ana Bernardo

M10137 - João Marques

Docente:

Professor Doutor Carlos Barrico

22 de dezembro de 2019

Resumo

Este documento relata o trabalho prático realizado na unidade curricular de Computação Científica do primeiro ano de mestrado em Engenharia Informática da Universidade da Beira Interior. Este trabalho permite-nos colocar em prática o que nos foi leccionado ao longo das aulas práticas assim como nas aulas teóricas, aprofundando deste modo os nossos conhecimentos.

O projeto consiste em implementar e desenvolver um modelo de simulação de um sistema relacionado com o funcionamento de um Serviço de Atendimento de uma Repartição de Finanças.

Este sistema de Finanças é constituído por 3 fases desde a chegada de um utente:

- **1ª Fase:** chegada à Repartição de Finanças com passagem pela zona de triagem;
- **2ª Fase:** passagem pela zona de balcões de atendimento específico de cada assunto;
- **3ª Fase:** passagem pela Tesouraria de Repartição de Finanças.

Lista de Figuras

2.1	Modelação do Serviço de Atendimento de uma Repartição de Fi- nanças	4
3.1	<i>Output</i> resultante da execução da função ilustrada na Listing 3.2 .	15
3.2	<i>Output</i> resultante da execução da função ilustrada na Listing 3.3 .	17
3.3	Número de utentes p/ período de tempo	17
3.4	<i>Output</i> Código do evento de chegada	22

Capítulo 1

Introdução

1.1 Objetivos

Este projeto tem como principais objectivos desenvolver um sistema relacionado com o funcionamento do Serviço de Atendimento de uma Repartição de Finanças para averiguar o desempenho da solução apresentada como enunciado. Este desempenho é medido pelos tempos de totais de espera mínimo , máximo e médio em todas as filas, quer nos períodos de tempo globais de 8h assim como também nos parciais (9-11h, 11-13h, 13-15h e 15-17h).

Também é necessário medir as taxas de ocupação de cada um dos postos de atendimento em cada um dos períodos de tempo parciais e globais.

1.2 Organização do Documento

De forma a demonstrar o trabalho executado, este documento foi estruturado da seguinte forma:

1. No primeiro capítulo – **Introdução** – apresenta os objetivos do trabalho e a respectiva organização do documento;
2. No segundo capítulo – **Modelação e Análise do Sistema** – apresenta uma explicação do serviço de atendimento de uma repartição de finanças com a afluência de utentes e análise do sistema através dos tempos de chegada e de ocupação nos respectivos balcões por onde os utentes passam;
3. No terceiro capítulo – **Simulação do Sistema** – apresenta o problema do enunciado do trabalho, o mecanismo de geração de números aleatórios e a construção do modelo de simulação;

4. No quarto capítulo – **Uso de estratégias para a resolução de problemas de desempenho nas pesquisas** –
5. Por fim, quinto capítulo – **Conclusão** – contém uma reflexão e uma análise geral ao trabalho desenvolvido.

Capítulo 2

Modelação e Análise do Sistema

2.1 Introdução

Neste capítulo iremos esquematizar e explicar o serviço de atendimento de uma repartição das finanças e as suas componentes/balcões e analisando a afluência de utentes no decorrer de um dia através dos seus tempos de chegada, e tempos de ocupação nos balcões por onde estes passam. Esta análise é essencial para a construção da simulação porque é baseado nesta análise que a simulação vai ser construída sendo que a lógica para a circulação dos utentes no serviço de atendimento irá ser descrita neste capítulo.

2.2 Modelação do Sistema

2.2.1 Especificações base de cada fase

Tal como é ilustrado na Figura 2.1, o sistema em questão é composto por três fases:

- *1.ª fase*

Na 1.ª fase do sistema existe apenas um posto de atendimento (Triagem) onde os utentes são canalizados para os respectivos balcões de atendimento, específicos de cada tipo de assunto (A, B ou C).

Verifica-se também a existência de uma fila para dois tipos de utentes: geral e prioritário. São considerados utentes prioritários, a título de exemplo, as grávidas e utentes com deficiências físicas.

Todos os utentes têm de passar por esta fase.

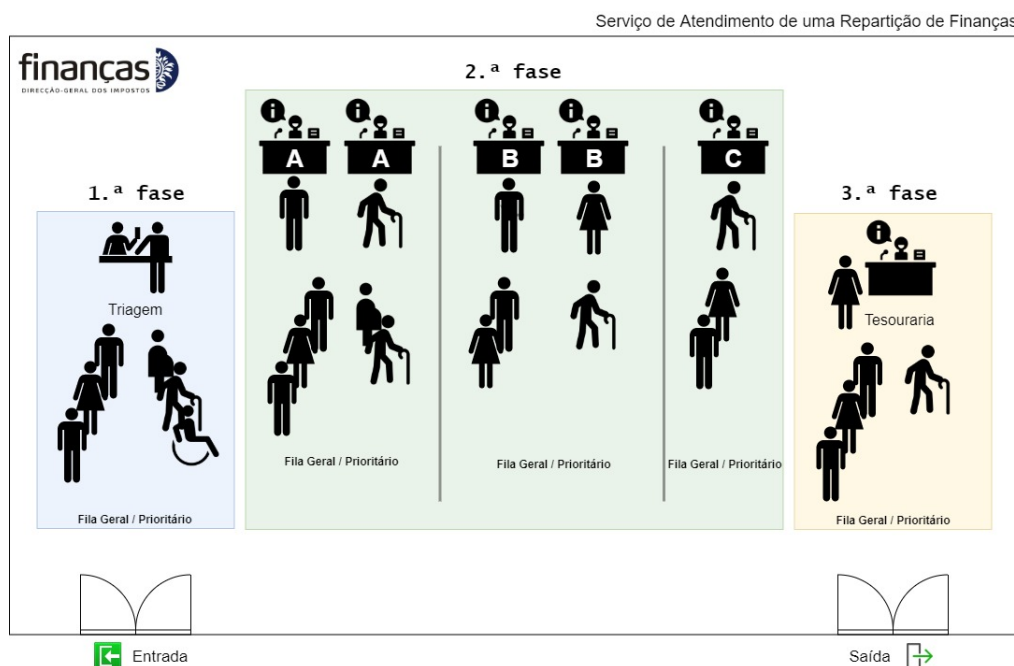


Figura 2.1: Modelação do Serviço de Atendimento de uma Repartição de Finanças

• 2.ª fase

Na 2.ª fase do sistema existem três tipos de assuntos a tratar nos balcões:

1. Assunto **A** – Certidões/Registos
2. Assunto **B** – IRS/IRC/IVA
3. Assunto **C** – Contencioso

Para tratar dos assuntos do tipo A, existem dois postos de atendimento com uma única fila. Para o assunto B existem também dois postos de atendimento com uma única fila. Para o assunto C existe apenas um posto de atendimento com única fila.

De salientar que todas as filas têm utentes prioritários e nem todos os utentes têm de passar por esta 2.ª fase.

• 3.ª fase

Na 3.ª fase do sistema existe um único balcão de atendimento para tratar de assuntos de Tesouraria. Existe uma fila, tendo prioridade os utentes considerados como tal. De referir que nem todos os utentes têm de passar por

esta fase e um utente depois de passar por esta fase, pode partir do sistema ou então regressar à 2.^a fase.

2.2.2 Modo de funcionamento

Um utente depois de passar pela 1.^a fase (Triagem) é encaminhado para a 2.^a fase ou diretamente para a 3.^a fase.

Se passar da 1.^a fase diretamente para a 3.^a fase, então depois de ser atendido nesta fase abandona o sistema, isto é, termina a sua passagem pelo sistema.

Se passar da 1.^a para a 2.^a fase, então podem acontecer três situações:

1. O utente após terminar o atendimento no respetivo balcão da 2.^a fase, abandona o sistema;
2. O utente após terminar o atendimento no respetivo balcão da 2.^a fase, passa para a 3.^a fase (Tesouraria) e, após terminar o seu atendimento na 3.^a fase, abandona o sistema;
3. O utente interrompe o seu atendimento no respetivo balcão da 2.^a fase, passa para a 3.^a fase (Tesouraria) e, após terminar o seu atendimento na 3.^a fase, regressa à 2.^a fase para continuar o seu atendimento no balcão onde interrompeu o atendimento. Após terminar de ser atendido no balcão da 2.^a fase, abandona o sistema.

No caso 3., o utente tem prioridade apenas sobre os utentes não prioritários ("geral"), mas não sobre os utentes "prioritários", quer na fila da 3.^a fase, quer na fila da 2.^a fase (mas neste caso apenas aquando do regresso).

2.3 Análise do Sistema

2.3.1 Características mínimas do sistema

Pretende-se analisar o sistema no período que decorre desde a abertura até ao fecho da Repartição das Finanças num dia de trabalho (8 horas), que é das 9h00 às 17h00. De modo a aumentar a granularidade da análise, considere-se o tempo total do horário laboral em segundos, ou seja, 8 horas = 480 minutos = 28800 segundos.

A Tabela 2.1 representa as taxas de variação de entradas dos clientes no sistema ao longo das 8 horas.

Tabela 2.1: Taxa de variação de entradas dos clientes no sistema

Período (Horas)	Taxa (%)
<i>09h00 - 11h00</i>	10
<i>11h00 - 13h00</i>	25
<i>13h00 - 15h00</i>	45
<i>15h00 - 17h00</i>	20

Na Tabela 2.2 pode observar-se a variação dos tempos de atendimento para o posto da 1.^a fase (Triagem).

Tabela 2.2: Taxa de variação dos tempos de atendimento no posto de Triagem

Período (Segundos)	Taxa (%)
<i>[0 - 60]</i>	55
<i>[60 - 120]</i>	35
<i>[120 - 180]</i>	10

A Tabela 2.3 representa as taxas de variação dos tempos de atendimento em cada um dos balcões da 2.^a fase, por tipo de assunto (A, B ou C).

Tabela 2.3: Taxa de variação dos tempos de atendimento nos balcões de atendimento da 2.^a fase

Período (Segundos)	Balcão (A) (%)	Balcão (B) (%)	Balcão (C) (%)
<i>[0 - 300]</i>	25	25	10
<i>[300 - 600]</i>	-	45	35
<i>[300 - 900]</i>	35	-	-
<i>[600 - 900]</i>	-	25	45
<i>[900 - 1200]</i>	-	5	10
<i>[900 - 1500]</i>	30	-	-
<i>[1500 - 1800]</i>	10	-	-

Por fim, a Tabela 2.4 representa a variação da taxa dos tempos de atendimento na 3.^a fase (Tesouraria).

Tabela 2.4: Taxa de variação dos tempos de atendimento no posto de Tesouraria

Período (Segundos)	Taxa (%)
[0 - 60]	40
[60 - 120]	55
[120 - 180]	5

São ainda conhecidas as taxas de variação respeitantes ao fluxo de utentes entre as três fases:

- **10%** dos utentes passa da 1.^a fase diretamente para a 3.^a fase;
- **20%** dos utentes são considerados prioritários logo na 1.^a fase, mantendo esta propriedade até abandonarem o sistema;
- **35%** dos utentes que passam pela 2.^a fase são atendidos nos postos do tipo A;
- **50%** dos utentes que passam pela 2.^a fase são atendidos nos postos do tipo B;
- **15%** dos utentes que passam pela 2.^a fase são atendidos nos postos do tipo C;
- **20%** dos utentes atendidos nos postos do tipo A passam pela 3.^a fase, sendo que **70%** destes abandonam o sistema logo a seguir, ou seja, apenas **30%** regressam à 2.^a fase;
- **30%** dos utentes atendidos nos postos do tipo B passam pela 3.^a fase, sendo que **80%** destes abandonam o sistema logo a seguir, ou seja, apenas **20%** regressam à 2.^a fase;
- **75%** dos utentes atendidos nos postos do tipo C passam pela 3.^a fase, sendo que **60%** destes abandonam o sistema logo a seguir, ou seja, apenas **40%** regressam à 2.^a fase.
- Passam pelo sistema (Repartição de Finanças), 120 a 150 utentes por dia, nas 8 horas de horário laboral.

2.4 Conclusão

Neste capítulo foram discutidas todas as especificações e o modo de funcionamento do serviço de atendimento de uma Repartição das Finanças. Tendo conhe-

cimento de todos os requisitos e também a forma como funciona o sistema, estão reunidas todas as condições para proceder à simulação do referido sistema.

Capítulo 3

Simulação do Sistema

3.1 Introdução

A simulação computacional de sistemas consiste na utilização de determinadas técnicas matemáticas, utilizadas em computadores digitais, as quais permitem imitar o funcionamento de praticamente qualquer tipo de operação ou processo (sistemas) do mundo real [1].

3.2 Modelos de Simulação de Sistemas

A simulação de sistemas pode ser dividida nas seguintes etapas básicas:

- **Problema:**

Identificação do problema apresentado pelo sistema em estudo e das partes (subsistemas) que interferem no problema.

No presente trabalho, pretende-se construir e implementar um modelo de simulação de um sistema relacionado com o funcionamento do Serviço de Atendimento de uma Repartição de Finanças, para averiguar o desempenho da solução na questão da medição dos tempos totais de espera em cada uma das filas.

- **Estudo do Sistema:**

Descrever o sistema em termos de componentes, atividades, entidades, eventos e restrições.

1. Entidades:

- ★ Utentes
- ★ Balcão Triagem
- ★ Posto Atend. A-1
- ★ Posto Atend. A-2
- ★ Posto Atend. B-1
- ★ Posto Atend. B-2
- ★ Posto Atend. C
- ★ Tesouraria

2. Atributos:

- ★ Número de Utentes
- ★ Tempo total de espera nas filas
- ★ Tempo total de espera nas filas (09h-11h)
- ★ Tempo total de espera nas filas (11h-13h)
- ★ Tempo total de espera nas filas (13h-15h)
- ★ Tempo total de espera nas filas (15h-17h)
- ★ Tempo total ocupação balcão Triagem
- ★ Tempo total ocupação Posto Atend. A-1
- ★ Tempo total ocupação Posto Atend. A-2
- ★ Tempo total ocupação Posto Atend. B-1
- ★ Tempo total ocupação Posto Atend. B-2
- ★ Tempo total ocupação Posto Atend. C
- ★ Tempo total ocupação balcão Tesouraria
- ★ Tempo total ocupação balcão Triagem (09h-11h)
- ★ Tempo total ocupação Posto Atend. A-1 (09h-11h)
- ★ Tempo total ocupação Posto Atend. A-2 (09h-11h)
- ★ Tempo total ocupação Posto Atend. B-1 (09h-11h)
- ★ Tempo total ocupação Posto Atend. B-2 (09h-11h)
- ★ Tempo total ocupação Posto Atend. C (09h-11h)
- ★ Tempo total ocupação balcão Tesouraria (09h-11h)
- ★ Tempo total ocupação balcão Triagem (11h-13h)

- ★ Tempo total ocupação Posto Atend. A-1 (11h-13h)
- ★ Tempo total ocupação Posto Atend. A-2 (11h-13h)
- ★ Tempo total ocupação Posto Atend. B-1 (11h-13h)
- ★ Tempo total ocupação Posto Atend. B-2 (11h-13h)
- ★ Tempo total ocupação Posto Atend. C (11h-13h)
- ★ Tempo total ocupação balcão Tesouraria (11h-13h)
- ★ Tempo total ocupação balcão Triagem (13h-15h)
- ★ Tempo total ocupação Posto Atend. A-1 (13h-15h)
- ★ Tempo total ocupação Posto Atend. A-2 (13h-15h)
- ★ Tempo total ocupação Posto Atend. B-1 (13h-15h)
- ★ Tempo total ocupação Posto Atend. B-2 (13h-15h)
- ★ Tempo total ocupação Posto Atend. C (13h-15h)
- ★ Tempo total ocupação balcão Tesouraria (13h-15h)
- ★ Tempo total ocupação balcão Triagem (15h-17h)
- ★ Tempo total ocupação Posto Atend. A-1 (15h-17h)
- ★ Tempo total ocupação Posto Atend. A-2 (15h-17h)
- ★ Tempo total ocupação Posto Atend. B-1 (15h-17h)
- ★ Tempo total ocupação Posto Atend. B-2 (15h-17h)
- ★ Tempo total ocupação Posto Atend. C (15h-17h)
- ★ Tempo total ocupação balcão Tesouraria (15h-17h)

3. Atividades:

4. Estado do Sistema:

- ★ Número do Utente
- ★ Tipo de Evento
- ★ Estado do Balcão da Triagem
- ★ Estado do Posto de Atend. A-1
- ★ Estado do Posto de Atend. A-2
- ★ Estado do Posto de Atend. B-1
- ★ Estado do Posto de Atend. B-2
- ★ Estado do Posto de Atend. C
- ★ Estado da Tesouraria

5. Eventos/Acontecimentos:

- ★ Evento de chegada à Repartição das Finanças
- ★ Evento de Partida da Triagem

- ★ Evento de Partida do Posto de Atend. A-1
- ★ Evento de Partida do Posto de Atend. A-2
- ★ Evento de Partida do Posto de Atend. B-1
- ★ Evento de Partida do Posto de Atend. B-2
- ★ Evento de Partida do Posto de Atend. C
- ★ Evento de Partida da Tesouraria

- **Modelo:**

Construção do modelo de representação do sistema em estudo, o qual precisa ser validado.

- **Solução:**

Realizar experiências sobre o modelo construído e utilizar as informações resultantes para propor soluções para o problema real.

- **Operacionalizar solução:**

Executar alterações no sistema real com base nos resultados da simulação.

3.3 Mecanismo de geração de números aleatórios

A geração de valores aleatórios para os tempos de chegada e para os tempos de atendimento num determinado posto socorreu-se do uso generalizado do Método de Monte Carlo.

3.3.1 Método de Monte Carlo

O método de Monte Carlo é um modelo de simulação estático usado para modelar fenômenos probabilísticos cujas características não mudam com o tempo. Este modelo necessita de números pseudoaleatórios, tal como os modelos de simulação dinâmicos, e pode também ser usado para avaliar expressões não probabilísticas através de métodos probabilísticos [1].

De modo geral, a simulação de Monte Carlo é definida por duas fases: *preparação* e *aplicação*. A fase de preparação consiste na recolha de dados, construção de tabelas com os intervalos de valores e as suas frequências. Ora, a informação proveniente desta primeira fase já é do nosso conhecimento, pelo que apenas tem de se seguir as etapas da segunda fase (aplicação), composta pelos seguintes passos:

1. Escolher um número $[0;1]$ através do uso de um programa ou algoritmo de geração de números aleatórios;

2. Procurar, na tabela de intervalos de frequência, o intervalo onde está o número escolhido;
3. O valor a ser usado na simulação é o ponto médio deste intervalo ou valor.

A única alteração feita à fase de aplicação foi no ponto 3., que em vez de o valor a ser usado corresponder ao ponto médio do intervalo, corresponde a um número aleatório dentro do intervalo de valores que consta na tabela. (Ex.: valor aleatório entre 0 e 7200, que corresponde a um período de 2 horas (09h – 11h).

Exemplo: Geração dos tempos de chegada para 100 utentes.

1.º passo – gerar 100 números aleatórios entre 0 e 1.

Utilizando uma função escrita em *Python*, o resultado seria:

```
# OBJETIVO: funcao que gera numeros entre 0 e 1 para um dado n.
#           de clientes
# PARAMETROS: seed --> Seed value is the previous value number
#              generated by the generator. (Ajustar p/ as %)
#              nClientes -> numero de clientes
# RETURN: lista com os nClientes numeros aleat rios entre 0 e 1
def getListNumerosEntre0e1(seed, nClientes):
    listNum01 = []

    # definir o valor de seed
    random.seed(seed)
    for x in range(0, nClientes):
        # gerar um numero aleatorio entre 0 e 1
        num = random.random()
        listNum01.append(num)

    return listNum01
```

Listing 3.1: Função em *Python* que devolve uma lista com números aleatórios entre 0 e 1

A função ilustrada na Listing 3.1 devolve uma lista com $nClientes$ valores entre 0 e 1. Para o exemplo apresentado, a variável $nClientes$ tomará o valor de 100. O valor do *seed* pode ser ajustado com base na coluna das frequências (%) correspondentes a cada intervalo de valores. Neste exemplo vamos assumir os valores da Tabela 2.1. O valor de 11 para a variável *seed* parece ser adequado, uma vez que a gama de valores obtidos com este valor de *seed* encaixam muito satisfatoriamente nas frequências (%) conhecidas e representadas na coluna "Taxa (%)" da Figura 2.1. Fazendo uma pequena modificação na função anterior, foi acrescentada uma variável *count* que guarda a quantidade de números entre 0 e 0.1 (10%) dos 100 números gerados e correspondente à taxa (%) no período das 09h00 às 11h00 = 2 horas = 7200 segundos. O valor da variável *count* deve rondar o valor 10 (10/100 = 10%).

```
def getListNumerosEntre0e1(seed, nClientes):
    listNum01 = []
    count=0

    # definir o valor de seed
    random.seed(seed)
    for x in range(0, nClientes):
        # gerar um numero aleatorio entre 0 e 1
        num = random.random()
        if (num >= 0 and num <= 0.1):
            count = count + 1

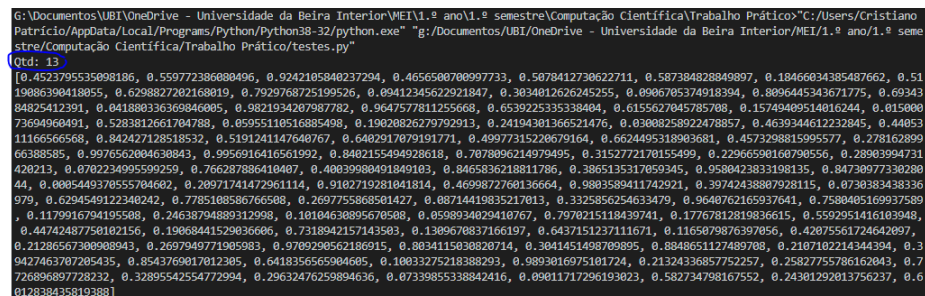
        listNum01.append(num)

    print(count)

    return listNum01
```

Listing 3.2: Função *getListNumerosEntre0e1* modificada

A execução da função apresentada na Listing 3.2 mostra o seguinte *output*:



```
G:\Documentos\UBI\OneDrive - Universidade da Beira Interior\VEI1.º ano\1.º semestre\Computação Científica\Trabalho Prático\C:\Users\Cristiano
Patricio\AppData\Local\Programs\Python\Python38-32\python.exe "g:\Documentos\UBI\OneDrive - Universidade da Beira Interior\VEI1.º ano\1.º seme
stre\Computação Científica\Trabalho Prático\testes.py"
QtD: 13
[0.4523795535098186, 0.559772386080496, 0.9242185840237294, 0.4656590700997733, 0.5078412730622711, 0.587304828049897, 0.18466034385487662, 0.51
19086390418855, 0.6298827282168819, 0.7929768725199526, 0.09412345622921847, 0.3834812626245255, 0.0906785374918394, 0.8996445343671775, 0.69343
84825412391, 0.041880336369846005, 0.9821934207987782, 0.9647577811255668, 0.6539225335338404, 0.6155627045785708, 0.15749409514016244, 0.015000
73694960491, 0.5283812661704788, 0.05955110516885498, 0.19020826279792913, 0.24194381366521476, 0.03008258922478857, 0.4639344612232845, 0.44053
11166566568, 0.842427128518532, 0.5191241147640767, 0.6402917079191771, 0.49977315220679164, 0.6624495318903681, 0.4573298815995577, 0.278162899
66388585, 0.9976562004638843, 0.9956916416561992, 0.8402155494928618, 0.7078096214979495, 0.3152772170155499, 0.22966590160790556, 0.28903994731
420213, 0.0702234995599259, 0.766287886410407, 0.40039980491849103, 0.8465836218811786, 0.3865135317059345, 0.9580423833198135, 0.84730977330280
44, 0.0005449370555704602, 0.20971741472961114, 0.9102719281041814, 0.4699872760136664, 0.9883589411742921, 0.39742438807928115, 0.0730383438336
979, 0.6294549122340242, 0.7785108586766508, 0.2697755868501427, 0.08714419835217013, 0.3325856254633479, 0.9640762165937641, 0.7580405169937589
, 0.1179916794195508, 0.24638794889312998, 0.10104630895670508, 0.0598934029410767, 0.7970215118439741, 0.17767812819836615, 0.5592951416103948,
0.44742487750182156, 0.19068441529036606, 0.7318942157143503, 0.1309670837166197, 0.6437151237111671, 0.1165079876397056, 0.42075561724642097,
0.21286567300908943, 0.2697949771905983, 0.9709290562186915, 0.0034115030820714, 0.3041451498709895, 0.8848651127489708, 0.2107102214344394, 0.3
9427463707205435, 0.8543769817012305, 0.6418356565904605, 0.10033275218388293, 0.9893016975101724, 0.21324336857752257, 0.25827755786162043, 0.7
726896897728232, 0.32895542554772994, 0.29632476259894636, 0.07339855338842416, 0.09011771296193023, 0.582734798167552, 0.24301292013756237, 0.6
012838435819388]
```

Figura 3.1: *Output* resultante da execução da função ilustrada na Listing 3.2

Como se pode observar na Figura 3.1 o valor obtido foi de **13**, que está próximo de 10. O objetivo não é conseguir obter um valor exatamente igual aos dados da amostra, mas sim próximo, de maneira a obter uma maior aleatoriedade para que a simulação se aproxime o mais possível da realidade.

2.º passo – gerar os tempos de chegada para os 100 clientes.

Tendo em linha de conta os valores obtidos (lista com números entre 0 e 1) pela função ilustrada na Listing 3.1, o 2.º passo é gerar aleatoriamente os tempos de chegada tendo em conta a frequência acumulada que se pode calcular a partir dos dados da Tabela 2.1. A Tabela 3.1 mostra a adição de mais duas colunas à Tabela 2.1.

Tabela 3.1: Reformulação da Tabela 2.1

Período (Horas)	Taxa (%)	Freq. Acum.	Intervalos
[09h00 - 11h00]	10	0.1	[0.0,0.1]
[11h00 - 13h00]	25	0.35	[0.11,0.35]
[13h00 - 15h00]	45	0.8	[0.36,0.80]
[15h00 - 17h00]	20	1	[0.81,1]

Tendo em conta a informação da Tabela acima referida, foi construída uma função em *Python* que permite obter os tempos de chegada para o 100 utentes.

```
# OBJETIVO: funcao que gera os tempos de chegada
# PARAMETROS: listNumbers --> lista de numeros aleatorios entre
#              0 e 1
# RETURN: lista com os tempos de chegada para os nClientes
def getTemposChegada(listNumbers):
    # lista c/ tempos das 9h -> 11h
    listInt1 = []
    # lista c/ tempos das 11h -> 13h
    listInt2 = []
    # lista c/ tempos das 13h -> 15h
    listInt3 = []
    # lista c/ tempos das 15h -> 17h
    listInt4 = []

    for num in listNumbers:
        if num >= 0 and num <= 0.1:
            # gerar um numero entre 0 e 7200
            listInt1.append(random.randrange(0,7200))
        elif num >= 0.11 and num <= 0.35:
            # gerar um numero entre 7201 e 14400
            listInt2.append(random.randrange(7201,14400))
        elif num >= 0.36 and num <= 0.80:
```

```

# gerar um numero entre 14401 e 21600
listInt3 . append(random . randrange(14401,21600))
else:
# gerar um numero entre 21601 e 28800
listInt4 . append(random . randrange(21601,28800))

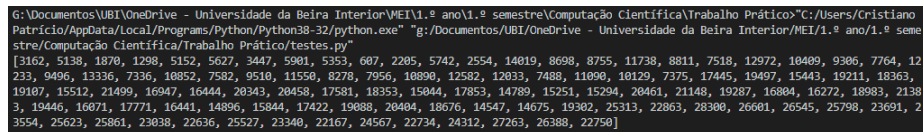
return listInt1 + listInt2 + listInt3 + listInt4

```

Listing 3.3: Função em *Python* que devolve uma lista com os tempos de chegada para os 100 utentes

A função anterior tem apenas um parâmetro de *input* (*listNumbers*), que é a lista obtida da execução da função *getListNumerosEntre0e1*.

O resultado da execução da função *getTemposChegada* é o seguinte:



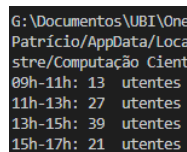
```

G:\Documentos\UBI\OneDrive - Universidade da Beira Interior\MEI\1.º ano\1.º semestre\Computação Científica\Trabalho Prático\C:\Users\Cristiano
Patricio\AppData\Local\Programs\Python\Python38-32\python.exe "g:\Documentos\UBI\OneDrive - Universidade da Beira Interior\MEI\1.º ano\1.º seme
stre\Computação Científica\Trabalho Prático\testes.py"
[3162, 5138, 1870, 1298, 5152, 5627, 3447, 5901, 5353, 607, 2285, 5742, 2554, 14819, 8698, 8755, 11738, 8811, 7518, 12872, 10409, 9306, 7764, 12
233, 9496, 13336, 7336, 10852, 7582, 9510, 11558, 8278, 7956, 10890, 12582, 12033, 7488, 11090, 10129, 7375, 17445, 19497, 15443, 19211, 18363,
19167, 15512, 21499, 16947, 16444, 20343, 20458, 17581, 18353, 15044, 17853, 14789, 15251, 15294, 20461, 21148, 19287, 16804, 16272, 18983, 2138
3, 19446, 16871, 17771, 16441, 14896, 15844, 17422, 19088, 20404, 18676, 14547, 14675, 19302, 25313, 22863, 28300, 26601, 26545, 25798, 23691, 2
3554, 25623, 25861, 23838, 22636, 25527, 23340, 22167, 24567, 22734, 24312, 27263, 26388, 22750]

```

Figura 3.2: *Output* resultante da execução da função ilustrada na Listing 3.3

Para comprovar se os valores obtidos encaixam e se aproximam da taxa(%) apresentada na Tabela 3.1, foi realizado um teste, em que foram contados os valores da lista obtida, respeitantes a cada intervalo (09h00-11h00; 11h00-13h00; 13h00-15h00; 15h00-17h00):



```

G:\Documentos\UBI\One
Patricio\AppData\Loca
stre\Computação Cient
09h-11h: 13 utentes
11h-13h: 27 utentes
13h-15h: 39 utentes
15h-17h: 21 utentes

```

Figura 3.3: Número de utentes p/ período de tempo

Tal como se pode observar na Figura 3.3, e comparando com a coluna "Taxa (%)" da Tabela 3.1, os valores andam ligeiramente próximos, considerando, claro está, 100 utentes, para que seja de fácil compreensão a relação entre a % e os resultados obtidos. A Tabela 3.2 resume os factos supracitados.

Tabela 3.2: Comparação em % das taxas calculadas e taxas obtidas para o número de utentes por intervalo de tempo

Período (Horas)	Taxa (%)	Taxa Obtida (%)
[09h00 - 11h00]	10	13
[11h00 - 13h00]	25	27
[13h00 - 15h00]	45	39
[15h00 - 17h00]	20	21

Foram gerados valores aleatórios para os tempos de chegada, tempos de atendimento na Triagem, tempos de atendimento no Posto A, B e C e também tempos de atendimento na Tesouraria. Todos estes valores foram gerados tendo em conta o que foi discutido anteriormente, seguindo a mesma lógica. Para facilitar o acesso aos dados, durante a simulação, foi criada uma classe **Utente** com os atributos:

- *tchegada* - (Tempo de Chegada)
- *tatendimento* - (Tempo de atendimento na Triagem)
- *tipoassunto* - (Tipo de Assunto - A, B, C ou '-')
- *tatend2fase1* - (Tempo de atendimento no balcão da 2.^a fase - 1.^a vez)
- *tatend2fase2* - (Tempo de atendimento no balcão da 2.^a fase - 2.^a vez)
- *tatend3fase* - (Tempo de atendimento no balcão da Tesouraria)

A Listing 3.4 mostra a classe **Utente**, construída em *Python*.

```
class Cliente:
    def __init__(cliente, id, tchegada, tatendimento, tipoassunto,
                 tatend2fase1, tatend2fase2, tatend3fase):
        cliente.id = id
        cliente.tchegada = tchegada
        cliente.tatendimento = tatendimento
        cliente.tipoassunto = tipoassunto
        cliente.tatend2fase1 = tatend2fase1
        cliente.tatend2fase2 = tatend2fase2
        cliente.tatend3fase = tatend3fase
```

Listing 3.4: Classe Cliente

De seguida, uma lista de nome **listaUtentes** foi populada com os valores obtidos da geração aleatório dos tempos. Por exemplo, a seguinte instrução mostra toda a informação do utente *10* do sistema:

```
print(listaUtentes[10].id)
print(listaUtentes[10].tchegada)
print(listaUtentes[10].tatendimento)
print(listaUtentes[10].tipoassunto)
print(listaUtentes[10].tatend2fase1)
print(listaUtentes[10].tatend2fase2)
print(listaUtentes[10].tatend3fase)
```

Tendo todos os tempos gerados e guardados na listaUtentes, estão reunidas as condições para proceder ao desenvolvimento dos algoritmos que serão o motor da simulação do sistema.

3.4 Construção do modelo de simulação

Para esta simulação, foi construída uma tabela muito semelhante às leccionadas nesta unidade curricular, nesta simulação a tabela resultante tem cerca de 19 colunas, nos seguintes subcapítulos iremos explicar o processo de geração desta tabela.

3.4.1 Variáveis de estado do sistema

A tabela resultante da simulação é constituído por 19 colunas:

1. Tempo atual
2. Evento
3. Cliente
4. Próxima chegada ao sistema
5. Fila de espera triagem
6. Estado da triagem
7. Próxima partida da triagem
8. Fila de espera do balcão A
9. Estado do balcão A
10. Próxima partida do balcão A
11. Fila de espera do balcão B

12. Estado do balcão B
13. Próxima partida do balcão B
14. Fila de espera do balcão C
15. Estado do balcão C
16. Próxima partida do balcão C
17. Fila de espera do balcão T
18. Estado do balcão T
19. Próxima partida do balcão T

O tempo atual é a variável do relógio de simulação, que regista em que tempo aconteceram os eventos do sistema. A coluna de evento diz que evento é que aconteceu no tempo atual enquanto a coluna cliente diz que cliente é que originou este evento. A coluna próxima chegada ao sistema regista em que tempo o próximo cliente chegará. As colunas 5, 8, 11, 14 e 17 correspondem a filas de espera dos vários balcões, as colunas 6, 9, 12, 15 e 18 são os estados dos vários balcões ("livre" ou "ocupado") e por fim as colunas 7, 10, 13, 16 e 19 são os tempos das próximas partidas dos vários balcões, o que irá originar eventos.

Além destas variáveis foram utilizadas outras como auxílio, estas variáveis são `proximo_cliente` contendo o id do próximo cliente a chegar ao sistema e `cliente_secretaria` que contém o id do utente atualmente a ocupar o posto de triagem. As variáveis `cliente_postoA`, `cliente_postoB`, `cliente_postoC` e `cliente_postoT` contêm o id dos utentes nos balcões A, B, C e T respectivamente. Por fim, como os utentes podem passar da segunda fase para a terceira e vice versa foi criada uma lista chamada `ciclo_infinito` que marca quando os utentes passam pela terceira fase, este variável tem o objectivo de prevenir haver um ciclo infinito já que os utentes apenas passam uma vez na terceira fase, ou seja, um cliente que está na segunda fase ao passar para a terceira, altera o valor na lista na posição do seu id e assim quando o utente regressa à segunda fase ao verificar esta lista o utente sai do sistema.

3.4.2 Relógio de simulação

O relógio de simulação assinala a que tempo foi feito cada evento. O próximo valor de relógio de simulação será o valor mínimo dos próximos eventos do sistema, esta variável é importante para assinalar o acontecimento destes eventos e controlar os tempos de espera sendo que o cálculo destes tempos de espera de cada

cliente é feito através da subtração do valor de relógio de simulação quando um cliente chegou a uma componente do sistema com o valor de relógio de simulação quando este mesmo utente entra num posto da componente do sistema.

3.4.3 Lista de eventos

Neste sistema há seis eventos possíveis:

1. Chegada ao sistema
2. Partida da triagem
3. Partida do balcão A
4. Partida do balcão B
5. Partida do balcão C
6. Partida do balcão T

Cada evento é diferenciado pelas três primeiras colunas de cada linha, ou seja cada linha tem uma combinação das três primeiras colunas diferente. O próximo evento a ocorrer é o que tiver o menor tempo no relógio de simulação e por isso os estes eventos podem ser explicados da seguinte maneira:

Próxima chegada ao sistema

Para este evento acontecer é preciso ter em conta a ultima linha da tabela, caso a coluna 4 tiver o menor valor entre os outros eventos.

Este evento apenas altera as primeiras 7 colunas em relação ao evento anterior caso a o estado da triagem esteja livre, caso esteja ocupado apenas altera as primeiras 5. As três primeiras colunas caracterização o evento enquanto na coluna 4 o valor da próxima chegada é alterado para o tempo de chegada do próximo cliente a chegar.

Se o estado do balcão de tesouraria for 'livre' a fila de espera da triagem continua vazia, o estado é alterado para 'ocupado', é calculado o tempo de próxima partida do balcão de triagem, através do tempo atual e do tempo de demora na triagem gerado, a variável de cliente_secretaria é alterada para o id deste utente. Caso o estado do balcão for 'ocupado' o cliente é adicionado à fila de espera e as colunas de estado e próxima partida da triagem não são alterados. Finalmente a variável de proximo_utente é alterado para o utente com o tempo de chegada mais baixo acima do tempo atual.

O código feito para o evento de chegada de um utente ao sistema é:


```

if (anterior[3]==1000 and anterior[6]==1000 and anterior[9]==1000 and anterior[12]==1000 and anterior[15]==1000 and anterior[18]==1000):
    return 0
else:
    linha = []
    if(anterior[3]<=anterior[6] and anterior[3]<=anterior[9] and anterior[3]<=anterior[12] and anterior[3]<anterior[15] and anterior[3]<=anterior[18]):
        linha.append(anterior[3])
        linha.append('Chegada')
        linha.append(proximo_cliente)
        prox=knock_knock(proximo_cliente)
        linha.append(prox[1])
        if(anterior[5]!='livre'):
            linha.append(anterior[4])
            linha.append('ocupado')
            linha.append(anterior[3]+ 'bye bye secretaria(proximo_cliente)')
            cliente_secretaria=proximo_cliente
        else:
            fila=insereir(anterior[4],proximo_cliente)
            linha.append(fila)
            linha.append('ocupado')
            linha.append(anterior[6])
        linha.append(anterior[7])
        linha.append(anterior[8])
        linha.append(anterior[9])
        linha.append(anterior[10])
        linha.append(anterior[11])
        linha.append(anterior[12])
        linha.append(anterior[13])
        linha.append(anterior[14])
        linha.append(anterior[15])
        linha.append(anterior[16])
        linha.append(anterior[17])
        linha.append(anterior[18])
        proximo_cliente=prox[0]

```

Figura 3.4: *Output* Código do evento de chegada

3.4.4 Contadores estatísticos

3.4.5 Rotinas

Rotina de inicialização

A rotina de inicialização consiste em inicializar todas as variáveis globais necessárias à simulação do sistema com os respectivos valores iniciais. Na Listing 3.5 pode observar-se a inicialização das referidas variáveis.

```

#Inicializa o de vari veis
INFINITO = 999999999
filaTriagem = []
filaPostoA = []
filaPostoB = []
filaPostoC = []
filaTesouraria = []
clock = 0
NUtente = ''
ETriagem = 0 #livre
TUFTriagem = 0
TATriagem = 0
TPChegada = listaUtentes[0].tchegada
TPTriagem = INFINITO
NUSistema = 0
TTEsperaTriagem = 0
TTEsperaTriagem0911 = 0
TTEsperaTriagem1113 = 0

```

```

TTesperaTriagem1315 = 0
TTesperaTriagem1517 = 0
TTOcupacaoTriagem = 0
TTOcupacaoTriagem0911 = 0
TTOcupacaoTriagem1113 = 0
TTOcupacaoTriagem1315 = 0
TTOcupacaoTriagem1517 = 0
nUtentes = 0
terminar = 0
UtentesTriagem = []
EPostoA1 = 0 #livre
EPostoA2 = 0 #livre
TUFPostoA = 0
TAPosttoA2 = 0
TPPostoA2 = INFINITO
TTOcupacaoPostoA2 = 0
TTOcupacaoPostoA20911 = 0
TTOcupacaoPostoA21113 = 0
TTOcupacaoPostoA21315 = 0
TTOcupacaoPostoA21517 = 0
TAPosttoA1 = 0
TPPostoA1 = INFINITO
TTOcupacaoPostoA1 = 0
TTOcupacaoPostoA10911 = 0
TTOcupacaoPostoA11113 = 0
TTOcupacaoPostoA11315 = 0
TTOcupacaoPostoA11517 = 0
TTesperaPostoA = 0
TTesperaPostoA0911 = 0
TTesperaPostoA1113 = 0
TTesperaPostoA1315 = 0
TTesperaPostoA1517 = 0
NUPostoA = 0
EPostoB1 = 0 #livre
EPostoB2 = 0 #livre
TUFPostoB = 0
TAPosttoB2 = 0
TPPostoB2 = INFINITO
TTOcupacaoPostoB2 = 0
TTOcupacaoPostoB20911 = 0
TTOcupacaoPostoB21113 = 0
TTOcupacaoPostoB21315 = 0
TTOcupacaoPostoB21517 = 0
TAPosttoB1 = 0
TPPostoB1 = INFINITO
TTOcupacaoPostoB1 = 0
TTOcupacaoPostoB10911 = 0
TTOcupacaoPostoB11113 = 0
TTOcupacaoPostoB11315 = 0

```

```
TTocupacaoPostoB11517 = 0
NUPostoB = 0
TTEsperaPostoB = 0
TTEsperaPostoB0911 = 0
TTEsperaPostoB1113 = 0
TTEsperaPostoB1315 = 0
TTEsperaPostoB1517 = 0
EPostoC = 0 #livre
TUFPostoC = 0
TAPostoc = 0
TPPostoC = INFINITO
TTocupacaoPostoC = 0
TTocupacaoPostoC0911 = 0
TTocupacaoPostoC1113 = 0
TTocupacaoPostoC1315 = 0
TTocupacaoPostoC1517 = 0
NUPostoC = 0
TTEsperaPostoC = 0
TTEsperaPostoC0911 = 0
TTEsperaPostoC1113 = 0
TTEsperaPostoC1315 = 0
TTEsperaPostoC1517 = 0
UtentesPostoC = []
UtentesPostoA1 = []
UtentesPostoA2 = []
UtentesPostoB1 = []
UtentesPostoB2 = []
TPTesouraria = INFINITO
ETesouraria = 0 #livre
TUFTesouraria = 0
TATesouraria = 0
TTocupacaoTesouraria = 0
TTocupacaoTesouraria0911 = 0
TTocupacaoTesouraria1113 = 0
TTocupacaoTesouraria1315 = 0
TTocupacaoTesouraria1517 = 0
NUTesouraria = 0
UtentesTesouraria = []
TTEsperaTesouraria = 0
TTEsperaTesouraria0911 = 0
TTEsperaTesouraria1113 = 0
TTEsperaTesouraria1315 = 0
TTEsperaTesouraria1517 = 0
filaPostoC2Vez = []
filaPostoB2Vez = []
filaPostoA2Vez = []
```

Listing 3.5: Inicialização de Variáveis

Quanto às nomenclaturas dos nomes atribuídos às variáveis, seguiu-se o seguinte padrão (exemplo para as variáveis que dizem respeito ao balcão da Triagem - 1.^a fase):

- **NUSistema** - (Número de Utentes no Sistema)
- **ETriagem** - (Estado do Balcão da Triagem - Livre ou Ocupado)
- **TATriagem** - (Tempo de Atendimento no Balcão da Triagem)
- **TPTriagem** - (Tempo de Partida do Balcão da Triagem)
- **TUFTriagem** - (Total de Utentes na Fila da Triagem)
- **TTEsperaTriagem** - (Tempo Total de Espera na Fila da Triagem)
- **TTEsperaTriagem0911** - (Tempo Total de Espera na Fila da Triagem das 09h00 às 11h00)
- **TTEsperaTriagem1113** - (Tempo Total de Espera na Fila da Triagem das 11h00 às 13h00)
- **TTEsperaTriagem1315** - (Tempo Total de Espera na Fila da Triagem das 13h00 às 15h00)
- **TTEsperaTriagem1517** - (Tempo Total de Espera na Fila da Triagem das 15h00 às 17h00)
- **TTOcupacaoTriagem** - (Tempo Total de Ocupação do Balcão da Triagem - tempo de serviço)
- **TTOcupacaoTriagem0911** - (Tempo Total de Ocupação do Balcão da Triagem das 09h00 às 11h00)
- **TTOcupacaoTriagem1113** - (Tempo Total de Ocupação do Balcão da Triagem das 11h00 às 13h00)
- **TTOcupacaoTriagem1315** - (Tempo Total de Ocupação do Balcão da Triagem das 13h00 às 15h00)
- **TTOcupacaoTriagem1517** - (Tempo Total de Ocupação do Balcão da Triagem das 15h00 às 17h00)

Rotina de gestão de tempo

A rotina de gestão de tempo é descrita no algoritmo da Listing 3.6. O objetivo desta rotina é devolver o tempo do próximo evento (*Clock*), bem como o tipo de evento que lhe está associado.

```
Gestao de Tempo(TempoProximaChegada , TempoPartidaTriagem ,
    TempoPartidaPostoA1 , TempoPartidaPostoA2 , TempoPartidaPostoB1
    , TempoPartidaPostoB2 , TempoPartidaPostoC ,
    TempoPartidaTesouraria):
    Devolve o minimo de {TempoProximaChegada ,
        TempoPartidaTriagem , TempoPartidaPostoA1 ,
        TempoPartidaPostoA2 , TempoPartidaPostoB1 ,
        TempoPartidaPostoB2 , TempoPartidaPostoC ,
        TempoPartidaTesouraria}
    Devolve o tipo de evento que lhe esta associado(chegada ,
        partidaTriagem , partidaPostoA1 , partidaPostoA2 ,
        partidaPostoB1 , partidaPostoB2 , partidaPostoC ,
        partidaTesouraria)
```

Listing 3.6: Algoritmo - Gestão de Tempo

Evento de Chegada ao Sistema

O algoritmo do evento associado à chegada de um Utente ao sistema, é descrito na Listing 3.7. O primeiro passo é marcar o próximo evento de chegada (Tempo da Próxima Chegada). Este valor é obtido fazendo uma consulta à Lista dos Utentes, no campo **tcchegada**: *listaUtentes[indice+1].tcchegada*.

De seguida, verifica-se se o balcão da Triagem está ocupado (ponto 2.). Caso esteja ocupado, então o utente é colocado na fila de espera (FilaTriagem), senão o utente é atendido no balcão e calcula-se o respetivo tempo de partida do balcão, bem como o valor de outras variáveis estatísticas (Tempo Total de Ocupação do balcão da Triagem, Número de Utentes do Sistema...) nos períodos de tempo parciais de 2h (09h0-11h00, 11h00-13h00, 13h00-15h00, 15h00-17h00) e no período de tempo global de 8h.

```
Evento de Chegada (clock , listaUtentes )
1. Marcar o proximo evento de chegada (TPCchegada)
2. Se (EstadoTriagem = ocupado) Entao
    2.1 FilaTriagem <- InserirFilaTriagem(idUtente , TempoChegada
        , TempoAtendimentoTriagem)
    2.2 TotalClientesFilaTriagem <- TotalClientesFilaTriagem + 1
3. Senao
    3.1 NumUtentesSistema <- NumUtentesSistema + 1
    3.2 EstadoTriagem = ocupado
    3.3 Lista UtentesTriagem <- Inserir utente que ocupa o
        balcao da Triagem
    3.4 TempoAtendimentoTriagem <- consultar o valor na
        listaUtentes
    3.5 TempoPartidaTriagem <- Clock + TempoAtendimentoTriagem
    3.6 TempoTotalOcupacaoTriagem <- TempoTotalOcupacaoTriagem +
        TempoAtendimentoTriagem
    3.7 Se (Clock >= 0 E Clock <= 7200) Entao
        3.7.1 TempoTotalOcupacaoTriagem0911 <-
            TempoTotalOcupacaoTriagem0911 + TempoAtendimentoTriagem
    3.8 Senao_Se (Clock > 7200 E Clock <= 14400) Entao
        3.8.1 TempoTotalOcupacaoTriagem1113 <-
            TempoTotalOcupacaoTriagem1113 + TempoAtendimentoTriagem
    3.9 Senao_Se (Clock > 14400 E Clock <= 21600) Entao
        3.9.1 TempoTotalOcupacaoTriagem1315 <-
            TempoTotalOcupacaoTriagem1315 + TempoAtendimentoTriagem
    3.10 Senao
        3.10.1 TempoTotalOcupacaoTriagem1517 <-
            TempoTotalOcupacaoTriagem1517 + TempoAtendimentoTriagem
    3.11 Fim_Se
4. Fim_Se
```

Listing 3.7: Algoritmo do Evento de Chegada ao Sistema

Evento de Partida do Balcão de Triagem

O algoritmo associado ao Evento de Partida do Balcão de Triagem é representado na Listing 3.8.

Depois de um utente abandonar o balcão da Triagem, podem suceder-se dois cenários:

- O utente vai para a 2.^a fase (Balcão A, B ou C)
- O utente vai para a 3.^a fase - Tesouraria

Desta forma, a lista *UtentesTriagem* armazena sempre o último utente que passou pelo balcão da Triagem. Caso não haja utentes na fila de espera para a Triagem (*FilaTriagem* = []), então o balcão fica livre e o tempo de partida é definido como indefinido - infinito. Se houver utentes na fila de espera, então verifica-se se há utentes prioritários na fila e estes, como o próprio nome indica, serão prioritários face a outros utentes que estejam na fila. Depois de selecionado o utente da fila, procede-se ao cálculo do tempo de espera (tempo atual - tempo de chegada do utente ao sistema), do tempo de partida e das variáveis estatísticas.

```

Evento de PartidaTriagem (Clock, listaUtentes)
1. Utente2Fase(indice) ← consultar valor na lista
   UtentesTriagem
2. Eliminar o utente da lista dos UtentesTriagem
3. Se (FilaTriagem = []) Entao
   3.1 EstadoTriagem ← 0
   3.2 TempoPartidaTriagem ← INFINITO
4. Senao
   4.1 Se (FilaTriagem Tem Utentes Prioritarios) Entao
     4.1.1 Selecionar o utente prioritario da FilaTriagem com
         menor indice
   4.2 Senao
     4.2.1 Selecionar o utente da frente da FilaTriagem
   4.3 TempoChegadaUtente ← consultar valor na lista da
       FilaTriagem
   4.4 TempoEsperaUtente ← Clock - TempoChegadaUtente
   4.5 NumUtentesSistema ← NumUtentesSistema + 1
   4.6 Lista UtentesTriagem ← Inserir utente que ocupa o
       balc o da Triagem
   4.7 TempoAtendimentoTriagem ← consultar o valor na lista
   4.8 TempoPartidaTriagem ← Clock + TempoAtendimentoTriagem
   4.9 FilaTriagem ← RemoverUtenteFilaTriagem(idUtente)
   4.10 TempoTotalEspera ← TempoTotalEspera +
       TempoEsperaUtente
   4.11 TempoTotalOcupacaoTriagem ← TempoTotalOcupacaoTriagem
       + TempoAtendimentoTriagem

```

```
4.12 Se (Clock >= 0 E Clock <= 7200) Entao
    4.12.1 TempoTotalOcupacaoTriagem0911 <-
        TempoTotalOcupacaoTriagem0911 + TempoAtendimentoTriagem
4.13 Senao_Se (Clock > 7200 E Clock <= 14400) Entao
    4.13.1 TempoTotalOcupacaoTriagem1113 <-
        TempoTotalOcupacaoTriagem1113 + TempoAtendimentoTriagem
4.14 Senao_Se (Clock > 14400 E Clock <= 21600) Entao
    4.14.1 TempoTotalOcupacaoTriagem1315 <-
        TempoTotalOcupacaoTriagem1315 + TempoAtendimentoTriagem
4.15 Senao
    4.15.1 TempoTotalOcupacaoTriagem1517 <-
        TempoTotalOcupacaoTriagem1517 + TempoAtendimentoTriagem
4.16 Fim_Se
5. Fim_Se
6. TipoAssuntoUtente2Fase <- consultar o valor na listaUtentes
   (indice)
7. Se (TipoAssunto <- A) Entao
    7.1 EventoChegadaPostoA(indice ,clock ,1)
8. Senao_Se (TipoAssunto <- B) Entao
    8.1 EventoChegadaPostoB(indice ,clock ,1)
9. Senao_Se (TipoAssunto <- C) Entao
    9.1 EventoChegadaPostoC(indice ,clock ,1)
10. Senao_Se (TipoAssunto <- '-' ) Entao
    10.1 Se (TempoAtendimentoTesouraria(Utente2Fase) != 0) Entao
        10.1.1 EventoChegadaTesouraria(indice ,clock ,1)
    10.2 Fim_Se
11. Fim_Se
```

Listing 3.8: Algoritmo do Evento de Partida do balcão de Triagem

Evento de Chegada ao Posto A

Tal como foi referido na secção anterior, um utente quando abandona o balcão de Triagem, pode dirigir-se a um balcão da 2.^a fase para tratar de um determinado tipo de assunto (A, B ou C), ou pode ir directamente á Tesouraria (3.^a fase) e depois abandonar o sistema. O algoritmo associado ao Evento de Chegada ao Posto A é o descrito na Listing 3.9.

De salientar que um utente que abandona a 2.^a fase e vai para a 3.^a fase (Tesouraria) ainda pode regressar uma segunda vez à 2.^a fase, pelo que este deve ser um aspeto a ter em atenção em termos de algoritmia.

Como se pode observar no cabeçalho do algoritmo, a variável **nvez**, passada como parâmetro, indica se o utente está a chegar ao balcão pela primeira vez (**nvez=1**) ou se vem pela segunda vez (**nvez=2**).

Deve ter-se ainda em atenção que o Posto A é constituído por dois balcões (PostoA1 e PostoA2). O que distingue a chegada de um utente pela primeira vez ou pela segunda vez é que na segunda vez o utente tem prioridade sobre os utentes que estão na fila do Posto A (exceto sobre os prioritários).

Em termos de algoritmo, a lógica é a mesma do Evento de Chegada ao Sistema, com a diferença de que neste caso existe mais um balcão de atendimento. Neste caso, verifica-se a disponibilidade de ambos os balcões e os utentes são escoados de acordo com essa disponibilidade. Caso os dois balcões (Posto A1 e Posto A2) estejam livres, o utente é atendido no Posto A1.

```

Evento de Chegada Posto A (indiceUtente , clock , nvez):
1. Se (EstadoPostoA1 = ocupado) Entao
  1.1. Se (EstadoPostoA2 = ocupado) Entao
    1.1.1 Se (nvez = 2) Entao
      1.1.1.1 FilaPostoA2Vez <- InserirFilaPostoA2Vez(idUtente
        , TempoChegada , tatendimento)
      1.1.1.2 TotalClientesFilaPostoA <-
        TotalClientesFilaPostoA + 1
    1.1.2 Senao
      1.1.2.1 FilaPostoA <- InserirFilaPostoA(idUtente ,
        TempoChegada)
      1.1.2.2 TotalClientesFilaPostoA <-
        TotalClientesFilaPostoA + 1
    1.1.3 Fim_Se
  1.2. Senao
    1.2.1 Se (nvez = 2) Entao
      1.2.1.1 EstadoPostoA2 = ocupado
      1.2.1.2 TempoAtendimentoPostoA2 <- consultar o valor na
        listaUtentes(indiceUtente)
      1.2.1.3 TempoPartidaPostoA2 <- Clock +
        TempoAtendimentoPostoA2

```

```

1.2.1.4 TempoTotalOcupacaoPostoA2 <-
    TempoTotalOcupacaoPostoA2 + TempoAtendimentoPostoA2
1.2.1.5 Se (Clock >= 0 E Clock <= 7200) Entao
    1.2.1.5.1 TempoTotalOcupacaoPostoA20911 <-
        TempoTotalOcupacaoPostoA20911 +
        TempoAtendimentoPostoA2
1.2.1.6 Senao_Se (Clock > 7200 E Clock <= 14400) Entao
    1.2.1.6.1 TempoTotalOcupacaoPostoA21113 <-
        TempoTotalOcupacaoPostoA21113 +
        TempoAtendimentoPostoA2
1.2.1.7 Senao_Se (Clock > 14400 E Clock <= 21600) Entao
    1.2.1.7.1 TempoTotalOcupacaoPostoA21315 <-
        TempoTotalOcupacaoPostoA21315 +
        TempoAtendimentoPostoA2
1.2.1.8 Senao
    1.2.1.8.1 TempoTotalOcupacaoPostoA21517 <-
        TempoTotalOcupacaoPostoA21517 +
        TempoAtendimentoPostoA2
1.2.1.9 Fim_Se
1.2.2 Senao
    1.2.2.1 EstadoPostoA2 = ocupado
    1.2.2.2 NumeroUtentesPostoA = NumeroUtentesPostoA + 1
    1.2.2.3 TempoAtendimentoPostoA2 <- consultar o valor na
        listaUtentes(indiceUtente)
    1.2.2.4 TempoPartidaPostoA2 <- Clock +
        TempoAtendimentoPostoA2
    1.2.2.5 TempoTotalOcupacaoPostoA2 <-
        TempoTotalOcupacaoPostoA2 + TempoAtendimentoPostoA2
    1.2.2.6 Se (Clock >= 0 E Clock <= 7200) Entao
        1.2.2.6.1 TempoTotalOcupacaoPostoA20911 <-
            TempoTotalOcupacaoPostoA20911 +
            TempoAtendimentoPostoA2
    1.2.2.7 Senao_Se (Clock > 7200 E Clock <= 14400) Entao
        1.2.2.7.1 TempoTotalOcupacaoPostoA21113 <-
            TempoTotalOcupacaoPostoA21113 +
            TempoAtendimentoPostoA2
    1.2.2.8 Senao_Se (Clock > 14400 E Clock <= 21600) Entao
        1.2.2.8.1 TempoTotalOcupacaoPostoA21315 <-
            TempoTotalOcupacaoPostoA21315 +
            TempoAtendimentoPostoA2
    1.2.2.9 Senao
        1.2.2.9.1 TempoTotalOcupacaoPostoA21517 <-
            TempoTotalOcupacaoPostoA21517 +
            TempoAtendimentoPostoA2
    1.2.2.10 Fim_Se
1.2.3 Fim_Se
1.3 Fim_Se
2. Senao
    2.1 Se (nvez = 2) Entao

```

```

2.1.1 EstadoPostoA1 = ocupado
2.1.2 TempoAtendimentoPostoA1 <- consultar o valor na
    listaUtentes(indiceUtente)
2.1.3 TempoPartidaPostoA1 <- Clock +
    TempoAtendimentoPostoA1
2.1.4 TempoTotalOcupacaoPostoA1 <-
    TempoTotalOcupacaoPostoA1 + TempoAtendimentoPostoA1
2.1.5 Se (Clock >= 0 E Clock <= 7200) Ent o
    2.1.5.1 TempoTotalOcupacaoPostoA10911 <-
        TempoTotalOcupacaoPostoA10911 +
        TempoAtendimentoPostoA1
2.1.6 Sen oSe (Clock > 7200 E Clock <= 14400) Ent o
    2.1.6.1 TempoTotalOcupacaoPostoA11113 <-
        TempoTotalOcupacaoPostoA11113 +
        TempoAtendimentoPostoA1
2.1.7 Sen oSe (Clock > 14400 E Clock <= 21600) Ent o
    2.1.7.1 TempoTotalOcupacaoPostoA11315 <-
        TempoTotalOcupacaoPostoA11315 +
        TempoAtendimentoPostoA1
2.1.8 Sen o
    2.1.8.1 TempoTotalOcupacaoPostoA11517 <-
        TempoTotalOcupacaoPostoA11517 +
        TempoAtendimentoPostoA1
2.1.9 Fim_Se
2.2 Senao
2.2.1 EstadoPostoA1 = ocupado
2.2.2 NumeroUtentesPostoA = NumeroUtentesPostoA + 1
2.2.3 TempoAtendimentoPostoA1 <- consultar o valor na
    listaUtentes(indiceUtente)
2.2.4 TempoPartidaPostoA1 <- Clock +
    TempoAtendimentoPostoA1
2.2.5 TempoTotalOcupacaoPostoA1 <-
    TempoTotalOcupacaoPostoA1 + TempoAtendimentoPostoA1
2.2.6 Se (Clock >= 0 E Clock <= 7200) Entao
    2.2.6.1 TempoTotalOcupacaoPostoA10911 <-
        TempoTotalOcupacaoPostoA10911 +
        TempoAtendimentoPostoA1
2.2.7 Senao_Se (Clock > 7200 E Clock <= 14400) Entao
    2.2.7.1 TempoTotalOcupacaoPostoA11113 <-
        TempoTotalOcupacaoPostoA11113 +
        TempoAtendimentoPostoA1
2.2.8 Senao_Se (Clock > 14400 E Clock <= 21600) Entao
    2.2.8.1 TempoTotalOcupacaoPostoA11315 <-
        TempoTotalOcupacaoPostoA11315 +
        TempoAtendimentoPostoA1
2.2.9 Senao
    2.2.9.1 TempoTotalOcupacaoPostoA11517 <-
        TempoTotalOcupacaoPostoA11517 +
        TempoAtendimentoPostoA1

```

```
2.2.10 Fim_Se  
2.3 Fim_Se  
3. Fim_Se
```

Listing 3.9: Algoritmo do Evento de Chegada ao Posto A

3.4.6 Evento de Partida do Posto A1

O algoritmo representado em 3.10 consiste no evento de Partida do Posto A1. Depois de um utente abandonar o posto A1, podem suceder-se dois acontecimentos:

- O utente vai para a 3.^a fase, a Tesouraria;
- O utente abandona o sistema.

Deste modo, caso não hajam clientes na fila de espera para o Posto A (FilaPostoA=[]), então este posto fica livre e o tempo da próxima partida é alterado para infinito. Caso existam clientes na fila de espera, então verifica-se se existem utentes prioritários na fila e estes serão prioritários face aos restantes utentes existentes na fila. Depois de seleccionado o utente, prossegue-se ao cálculo do tempo de espera (tempo atual - tempo de chegada do utente ao sistema), do tempo de partida e das variáveis estatísticas.

Evento de Partida Posto A1 (clock):

1. Ute3Fase(indice) ← consultar valor na lista UtePostoA1
2. Eliminar o utente da lista dos UtePostoA1
3. Se (FilaPostoA = []) Entao
 - 3.1 Se (FilaPostoA2Vez == []) Entao
 - 3.1.1 EstadoPostoA1 ← livre
 - 3.1.2 TempoPartidaPostoA1 ← INFINITO
 - 3.2 Senao
 - 3.2.1 Se (FilaPostoA2Vez Tem Ute Prioritarios) Entao
 - 3.2.1.1 Selecionar o utente prioritario da lista com menor indice
 - 3.2.2 Senao
 - 3.2.2.1 Selecionar o utente da frente da fila
 - 3.2.3 TempoChegadaUte ← consultar valor na lista da FilaPostoA2Vez
 - 3.2.4 NumeroUtePostoA ← NumeroUtePostoA + 1
 - 3.2.5 TempoEsperaUte ← Clock - TempoChegadaUte
 - 3.2.6 TempoAtendimentoPostoA1 ← consultar o valor na lista da FilaPostoA2Vez
 - 3.2.7 TempoPartidaPostoA1 ← Clock + TempoAtendimentoPostoA1
 - 3.2.8 FilaPostoA2Vez ← RemoverUteFilaPostoA2Vez(idUte)
 - 3.2.9 TempoTotalEsperaFilaPostoA ← TempoTotalEsperaFilaPostoA + TempoEsperaUte
 - 3.2.10 TempoTotalOcupacaoPostoA1 ← TempoTotalOcupacaoPostoA1 + TempoAtendimentoPostoA1
 - 3.2.11 Se (Clock >= 0 E Clock <= 7200) Entao

```

3.2.11.1 TempoTotalOcupacaoPostoA10911 <-
    TempoTotalOcupacaoPostoA10911 +
    TempoAtendimentoPostoA1
3.2.12 Senao_Se (Clock > 7200 E Clock <= 14400) Entao
    3.2.12.1 TempoTotalOcupacaoPostoA11113 <-
        TempoTotalOcupacaoPostoA11113 +
        TempoAtendimentoPostoA1
3.2.13 Senao_Se (Clock > 14400 E Clock <= 21600) Entao
    3.2.13.1 TempoTotalOcupacaoPostoA11315 <-
        TempoTotalOcupacaoPostoA11315 +
        TempoAtendimentoPostoA1
3.2.14 Senao
    3.2.14.1 TempoTotalOcupacaoPostoA11517 <-
        TempoTotalOcupacaoPostoA11517 +
        TempoAtendimentoPostoA1
3.2.15 Fim_Se
3.3 Fim_Se
4. Senao
4.1 Se (FilaPostoA2Vez == []) Entao
    4.1.1 Se (FilaPostoA Tem Utentes Prioritarios) Entao
        4.1.1.1 Selecionar o utente prioritario da lista com
            menor indice
    4.1.2 Senao
        4.1.2.1 Selecionar o utente da frente da fila
    4.1.3 Fim_Se
    4.1.4 TempoChegadaUtente <- consultar valor na lista da
        FilaPostoA
    4.1.5 NumeroUtentesPostoA <- NumeroUtentesPostoA + 1
    4.1.6 TempoEsperaUtente <- Clock - TempoChegadaUtente
    4.1.7 TempoAtendimentoPostoA1 <- consultar o valor na
        lista da FilaPostoA
    4.1.8 TempoPartidaPostoA1 <- Clock +
        TempoAtendimentoPostoA1
    4.1.9 FilaPostoA <- RemoverUtenteFilaPostoA(idUtente)
    4.1.10 TempoTotalEsperaFilaPostoA <-
        TempoTotalEsperaFilaPostoA + TempoEsperaUtente
    4.1.11 TempoTotalOcupacaoPostoA1 <-
        TempoTotalOcupacaoPostoA1 + TempoAtendimentoPostoA1
    4.1.12 Se (Clock >= 0 E Clock <= 7200) Entao
        4.1.12.1 TempoTotalOcupacaoPostoA10911 <-
            TempoTotalOcupacaoPostoA10911 +
            TempoAtendimentoPostoA1
    4.1.13 Senao_Se (Clock > 7200 E Clock <= 14400) Entao
        4.1.13.1 TempoTotalOcupacaoPostoA11113 <-
            TempoTotalOcupacaoPostoA11113 +
            TempoAtendimentoPostoA1
    4.1.14 Senao_Se (Clock > 14400 E Clock <= 21600) Entao
        4.1.14.1 TempoTotalOcupacaoPostoA11315 <-
            TempoTotalOcupacaoPostoA11315 +

```

```
TempoAtendimentoPostoA1
4.1.15 Senao
    4.1.15.1 TempoTotalOcupacaoPostoA11517 <-
        TempoTotalOcupacaoPostoA11517 +
        TempoAtendimentoPostoA1
4.1.16 Fim_Se
4.2 Senao
    4.2.1 Se (FilaPostoA Tem Utentes Prioritarios) Entao
        4.2.1.1 Selecionar o utente prioritario com menor indice
        4.2.1.2 Voltar ao ponto 4.1.4
    4.2.2 Senao
        4.2.2.1 Voltar ao ponto 3.2.1
    4.2.3 Fim_Se
4.3 Fim_Se
5. Fim_Se
6. Se (TempoAtendimentoTesouraria(Utente3Fase) != 0) Entao
    6.1 EventoChegadaTesouraria(indice , clock)
7. Fim_Se
```

Listing 3.10: Evento de Partida do Posto A1

3.4.7 Evento de Partida do Posto A2

O pseudocódigo apresentado em 3.11 apresenta o Evento de Partida do Posto A2. Este evento é semelhante com o anterior, na qual se pode, suceder dois cenários:

- O utente parte para a 3.^a fase, a Tesouraria;
- O utente abandona o sistema;

Neste caso, se não existirem utentes na fila de espera para o Posto A (FilaPostoA = []), então este posto fica livre e o tempo da próxima partida é alterado para infinito. Caso existam clientes à espera para o balcão A, verifica-se na fila se existem utentes prioritários, se existirem, estes terão prioridade sobre os outros presentes na fila de espera. Depois de selecionado o utente, prossegue-se ao cálculo do tempo de espera, do tempo da próxima partida e das restantes variáveis estatísticas.

```
Evento de Partida Posto A2 (clock):
1. Utente3Fase(indice) <- consultar valor na lista
   UtentesPostoA2
2. Eliminar o utente da lista dos UtentesPostoA2
3. Se (FilaPostoA = []) Entao
3.1 Se (FilaPostoA2Vez = []) Entao
3.1.1 EstadoPostoA2 <- livre
3.1.2 TempoPartidaPostoA2 <- INFINITO
3.2 Senao
3.2.1 Se (FilaPostoA2Vez Tem Utentes Prioritarios) Entao
3.2.1.1 Selecionar o utente prioritario da lista com
      menor indice
3.2.2 Senao
3.2.2.1 Selecionar o utente da frente da fila
3.2.3 FIm_Se
3.2.4 TempoChegadaUtente <- consultar valor na lista da
      FilaPostoA2Vez
3.2.5 NumeroUtentesPostoA <- NumeroUtentesPostoA + 1
3.2.6 TempoEsperaUtente <- Clock - TempoChegadaUtente
3.2.7 TempoAtendimentoPostoA2 <- consultar o valor na
      lista FilaPostoA2Vez
3.2.8 TempoPartidaPostoA2 <- Clock +
      TempoAtendimentoPostoA2
3.2.9 FilaPostoA2Vez <- RemoverUtenteFilaPostoA2Vez(
      idUtente)
3.2.10 TempoTotalEsperaFilaPostoA <-
      TempoTotalEsperaFilaPostoA + TempoEsperaUtente
3.2.11 TempoTotalOcupacaoPostoA2 <-
      TempoTotalOcupacaoPostoA2 + TempoAtendimentoPostoA2
3.2.12 Se (Clock >= 0 E Clock <= 7200) Entao
```



```

3.2.12.1 TempoTotalOcupacaoPostoA20911 <-
    TempoTotalOcupacaoPostoA20911 +
    TempoAtendimentoPostoA2
3.2.13 Senao_Se (Clock > 7200 E Clock <= 14400) Entao
3.2.13.1 TempoTotalOcupacaoPostoA21113 <-
    TempoTotalOcupacaoPostoA21113 +
    TempoAtendimentoPostoA2
3.2.14 Senao_Se (Clock > 14400 E Clock <= 21600) Entao
3.2.14.1 TempoTotalOcupacaoPostoA21315 <-
    TempoTotalOcupacaoPostoA21315 +
    TempoAtendimentoPostoA2
3.2.15 Senao
3.2.15.1 TempoTotalOcupacaoPostoA21517 <-
    TempoTotalOcupacaoPostoA21517 +
    TempoAtendimentoPostoA2
3.2.16 Fim_Se
3.3 Fim_Se
4. Senao
4.1 Se (FilaPostoA2Vez = []) Entao
4.1.1 Se (FilaPostoA Tem Utentes Prioritarios) Entao
4.1.1.1 Selecionar o utente prioritario da lista com
    menor indice
4.2.1 Senao
4.2.1.1 Selecionar o utente da frente da fila
4.2.3 TempoChegadaUtente <- consultar valor na lista da
    FilaPostoA
4.2.4 NumeroUtentesPostoA <- NumeroUtentesPostoA + 1
4.2.5 TempoEsperaUtente <- Clock - TempoChegadaUtente
4.2.6 TempoAtendimentoPostoA2 <- consultar o valor na
    lista
4.2.7 TempoPartidaPostoA2 <- Clock +
    TempoAtendimentoPostoA2
4.2.8 FilaPostoA <- RemoverUtenteFilaPostoA(idUtente)
4.2.9 TempoTotalEsperaFilaPostoA <-
    TempoTotalEsperaFilaPostoA + TempoEsperaUtente
4.2.10 TempoTotalOcupacaoPostoA2 <-
    TempoTotalOcupacaoPostoA2 + TempoAtendimentoPostoA2
4.2.11 Se (Clock >= 0 E Clock <= 7200) Entao
4.2.11.1 TempoTotalOcupacaoPostoA20911 <-
    TempoTotalOcupacaoPostoA20911 +
    TempoAtendimentoPostoA2
4.2.12 Senao_Se (Clock > 7200 E Clock <= 14400) Entao
4.2.12.1 TempoTotalOcupacaoPostoA21113 <-
    TempoTotalOcupacaoPostoA21113 +
    TempoAtendimentoPostoA2
4.2.13 Senao_Se (Clock > 14400 E Clock <= 21600) Entao
4.2.13.1 TempoTotalOcupacaoPostoA21315 <-
    TempoTotalOcupacaoPostoA21315 +
    TempoAtendimentoPostoA2

```

```
4.2.14 Senao
  4.2.14.1 TempoTotalOcupacaoPostoA21517 <-
    TempoTotalOcupacaoPostoA21517 +
    TempoAtendimentoPostoA2
4.2.15 Fim_Se
4.2 Senao
  4.2.1 Se (FilaPostoA Tem Utentes Prioritarios) Entao
    4.2.1.1 Selecionar o utente prioritario com menor indice
    4.2.1.2 Voltar ao ponto 4.1.4
  4.2.2 Senao
    4.2.2.1 Voltar ao ponto 3.2.1
  4.2.3 Fim_Se
4.3 Fim_Se
5. Fim_Se
6. Se (TempoAtendimentoTesouraria(Utente3Fase) != 0) Entao
  6.1 EventoChegadaTesouraria(indice , clock)
7. Fim_Se
```

Listing 3.11: Evento de Partida do Posto A2

3.4.8 Evento de Chegada ao Posto B

Como foi referido anteriormente quando um utente abandona o balcão da triagem, pode dirigir-se para um dos balcões da 2.^a fase para tratar de um determinado tipo de assunto (A, B ou C) ou dirigir-se para a tesouraria (3.^a fase) e seguidamente abandonar o sistema. O algoritmo apresentado em 3.12 representa o Evento de Chegada ao posto B. Quando um cliente abandona a 2.^a fase e vai para a 3.^a fase ainda pode regressar novamente à 2.^a fase, pelo que se deve ter em consideração esta situação no algoritmo.

A variável **nvez** apresentada no cabeçalho do algoritmo indica se um utente está a chegar pela primeira vez (**nvez=1**) ou se vem pela segunda vez (**nvez=2**). Deve-se ter ainda em atenção que o Posto B é constituído por dois balcões (PostoB1 e PostoA2). O que diferencia a chegada de um utente pela primeira vez ou pela segunda vez é que na segunda vez tem prioridade sobre os utentes que estão na fila do Posto B com a excepção dos prioritários.

Em termos de algoritmo, a lógica é a mesma ao Evento de chegada ao Posto A. Deve-se verificar a disponibilidade de cada um dos balcões e os utentes são saindo de acordo com a disponibilidade dos balcões. Caso os dois balcões (Posto B1 e Posto B2) estejam livre, o cliente é atendido no Posto A1.

```

Evento de Chegada Posto B (indiceUtente , clock , nvez):
1. Se (EstadoPostoB1 = ocupado) Entao
  1.1 Se (EstadoPostoB2 = ocupado) Entao
    1.1.1 Se (nvez = 2) Entao
      1.1.1.1 FilaPostoB2Vez <- InserirFilaPostoB2Vez(idUtente
        , TempoChegada ,tatendimento)
      1.1.1.2 TotalClientesFilaPostoB <-
        TotalClientesFilaPostoB + 1
    1.1.2 Senao
      1.1.2.1 FilaPostoB <- InserirFilaPostoB(idUtente ,
        TempoChegada)
      1.1.2.2 TotalClientesFilaPostoB <-
        TotalClientesFilaPostoB + 1
    1.1.3 Fim_Se
  1.2 Senao
    1.2.1 Se (nvez = 2) Entao
      1.2.1.1 EstadoPostoB2 = ocupado
      1.2.1.2 TempoAtendimentoPostoB2 <- consultar o valor na
        listaUtentes(indiceUtente)
      1.2.1.3 TempoPartidaPostoB2 <- Clock +
        TempoAtendimentoPostoB2
      1.2.1.4 TempoTotalOcupacaoPostoB2 <-
        TempoTotalOcupacaoPostoB2 + TempoAtendimentoPostoB2
      1.2.1.5 Se (Clock >= 0 E Clock <= 7200) Entao
        1.2.1.5.1 TempoTotalOcupacaoPostoB20911 <-

```

```

        TempoTotalOcupacaoPostoB20911 +
        TempoAtendimentoPostoB2
1.2.1.6 Senao_Se (Clock > 7200 E Clock <= 14400) Entao
    1.2.1.6.1 TempoTotalOcupacaoPostoB21113 <-
        TempoTotalOcupacaoPostoB21113 +
        TempoAtendimentoPostoB2
1.2.1.7 Senao_Se (Clock > 14400 E Clock <= 21600) Entao
    1.2.1.7.1 TempoTotalOcupacaoPostoB21315 <-
        TempoTotalOcupacaoPostoB21315 +
        TempoAtendimentoPostoB2
1.2.1.8 Senao
    1.2.1.8.1 TempoTotalOcupacaoPostoB21517 <-
        TempoTotalOcupacaoPostoB21517 +
        TempoAtendimentoPostoB2
1.2.1.9 Fim_Se
1.2.2 Senao
    1.2.2.1 EstadoPostoB2 = ocupado
    1.2.2.2 NumeroUtentesPostoB = NumeroUtentesPostoB + 1
    1.2.2.3 TempoAtendimentoPostoB2 <- consultar o valor na
        listaUtentes(indiceUtente)
    1.2.2.4 TempoPartidaPostoB2 <- Clock +
        TempoAtendimentoPostoB2
    1.2.2.5 TempoTotalOcupacaoPostoB2 <-
        TempoTotalOcupacaoPostoB2 + TempoAtendimentoPostoB2
    1.2.2.6 Se (Clock >= 0 E Clock <= 7200) Entao
        1.2.2.6.1 TempoTotalOcupacaoPostoB20911 <-
            TempoTotalOcupacaoPostoB20911 +
            TempoAtendimentoPostoB2
    1.2.2.7 Senao_Se (Clock > 7200 E Clock <= 14400) Entao
        1.2.2.7.1 TempoTotalOcupacaoPostoB21113 <-
            TempoTotalOcupacaoPostoB21113 +
            TempoAtendimentoPostoB2
    1.2.2.8 Senao_Se (Clock > 14400 E Clock <= 21600) Entao
        1.2.2.8.1 TempoTotalOcupacaoPostoB21315 <-
            TempoTotalOcupacaoPostoB21315 +
            TempoAtendimentoPostoB2
    1.2.2.9 Senao
        1.2.2.9.1 TempoTotalOcupacaoPostoB21517 <-
            TempoTotalOcupacaoPostoB21517 +
            TempoAtendimentoPostoB2
    1.2.2.10 Fim_Se
1.2.3 Fim_Se
1.3 Fim_Se
2. Senao
    2.1 Se (nvez = 2) Entao
        2.1.1 EstadoPostoB1 = ocupado
        2.1.2 TempoAtendimentoPostoB1 <- consultar o valor na
            listaUtentes(indiceUtente)
        2.1.3 TempoPartidaPostoB1 <- Clock +

```

```

    TempoAtendimentoPostoB1
2.1.4 TempoTotalOcupacaoPostoB1 <-
    TempoTotalOcupacaoPostoB1 + TempoAtendimentoPostoB1
2.1.5 Se (Clock >= 0 E Clock <= 7200) Entao
    2.1.5.1 TempoTotalOcupacaoPostoB10911 <-
        TempoTotalOcupacaoPostoB10911 +
        TempoAtendimentoPostoB1
2.1.6 Senao_Se (Clock > 7200 E Clock <= 14400) Entao
    2.1.6.1 TempoTotalOcupacaoPostoB11113 <-
        TempoTotalOcupacaoPostoB11113 +
        TempoAtendimentoPostoB1
2.1.7 Senao_Se (Clock > 14400 E Clock <= 21600) Entao
    2.1.7.1 TempoTotalOcupacaoPostoB11315 <-
        TempoTotalOcupacaoPostoB11315 +
        TempoAtendimentoPostoB1
2.1.8 Senao
    2.1.8.1 TempoTotalOcupacaoPostoB11517 <-
        TempoTotalOcupacaoPostoB11517 +
        TempoAtendimentoPostoB1
2.1.9 Fim_Se
2.2 Senao
2.2.1 EstadoPostoB1 = ocupado
2.2.2 NumeroUtentesPostoB = NumeroUtentesPostoB + 1
2.2.3 TempoAtendimentoPostoB1 <- consultar o valor na
    listaUtentes(indiceUtente)
2.2.4 TempoPartidaPostoB1 <- Clock +
    TempoAtendimentoPostoB1
2.2.5 TempoTotalOcupacaoPostoB1 <-
    TempoTotalOcupacaoPostoB1 + TempoAtendimentoPostoB1
2.2.6 Se (Clock >= 0 E Clock <= 7200) Entao
    2.2.6.1 TempoTotalOcupacaoPostoB10911 <-
        TempoTotalOcupacaoPostoB10911 +
        TempoAtendimentoPostoB1
2.2.7 Senao_Se (Clock > 7200 E Clock <= 14400) Entao
    2.2.7.1 TempoTotalOcupacaoPostoB11113 <-
        TempoTotalOcupacaoPostoB11113 +
        TempoAtendimentoPostoB1
2.2.8 Senao_Se (Clock > 14400 E Clock <= 21600) Entao
    2.2.8.1 TempoTotalOcupacaoPostoB11315 <-
        TempoTotalOcupacaoPostoB11315 +
        TempoAtendimentoPostoB1
2.2.9 Senao
    2.2.9.1 TempoTotalOcupacaoPostoB11517 <-
        TempoTotalOcupacaoPostoB11517 +
        TempoAtendimentoPostoB1
2.2.10 Fim_Se
2.3 Fim_Se
3. Fim_Se

```

Listing 3.12: Algoritmo do Evento de Chegada ao Posto B

3.4.9 Evento de Partida do Posto B1

O algoritmo representado em 3.13 consiste no evento de Partida do Posto B1. Depois de um utente abandonar o posto B1, podem suceder-se dois cenários:

- O utente abandona o sistema;
- O utente segue para a 3.^a fase, a Tesouraria;

Desta forma, se não existir nenhum utente na fila de espera para o Posto B (FilaPostoB=[]), o que indica que o posto fica livre e o tempo da próxima partida é modificado para indeterminado ou seja infinito. Caso existam clientes na fila de espera, verifica-se se há utentes prioritários na fila, e estes como o nome indica têm prioridade sobre os restantes utentes na fila de espera. Depois de seleccionado o utente, segue-se para o cálculo do tempo de espera (tempo atual - tempo de chegada do utente ao sistema), do tempo de partida e das variáveis estatísticas.

Evento de Partida Posto B1 (clock):

1. Utente3Fase(indice) ← consultar valor na lista UtentesPostoB1
2. Eliminar o utente da lista dos UtentesPostoB1
3. Se (FilaPostoB = []) Entao
 - 3.1 Se (FilaPostoB2Vez == []) Entao
 - 3.1.1 EstadoPostoB1 ← livre
 - 3.2.1 TempoPartidaPostoB1 ← INFINITO
 - 3.2 Senao
 - 3.2.1 Se (FilaPostoB2Vez Tem Uteses Prioritarios) Entao
 - 3.2.1.1 Selecionar o utente prioritario da lista com menor indice
 - 3.2.2 Senao
 - 3.2.2.1 Selecionar o utente da frente da fila
 - 3.2.3 Fim_Se
 - 3.2.4 TempoChegadaUtente ← consultar valor na lista da FilaPostoB2Vez
 - 3.2.5 TempoEsperaUtente ← Clock – TempoChegadaUtente
 - 3.2.6 TempoAtendimentoPostoB ← consultar o valor na lista da FilaPostoB2Vez
 - 3.2.7 TempoPartidaPostoB ← Clock + TempoAtendimentoPostoB
 - 3.2.8 FilaPostoB2Vez ← RemoverUtenteFilaPostoB2Vez(idUtente)
 - 3.2.9 TempoTotalEsperaFilaPostoB ← TempoTotalEsperaFilaPostoB + TempoEsperaUtente
 - 3.2.10 TempoTotalOcupacaoPostoB ← TempoTotalOcupacaoPostoB + TempoAtendimentoPostoB
 - 3.2.11 Se (Clock >= 0 E Clock <= 7200) Entao
 - 3.2.11.1 TempoTotalOcupacaoPostoB0911 ← TempoTotalOcupacaoPostoB0911 + TempoAtendimentoPostoB

```

3.2.12 Senao_Se (Clock > 7200 E Clock <= 14400) Entao
    3.2.12.1 TempoTotalOcupacaoPostoB1113 <-
        TempoTotalOcupacaoPostoB1113 + TempoAtendimentoPostoB
3.2.13 Senao_Se (Clock > 14400 E Clock <= 21600) Entao
    3.2.13.1 TempoTotalOcupacaoPostoB1315 <-
        TempoTotalOcupacaoPostoB1315 + TempoAtendimentoPostoB
3.2.14 Senao
    3.2.14.1 TempoTotalOcupacaoPostoB1517 <-
        TempoTotalOcupacaoPostoB1517 + TempoAtendimentoPostoB
3.2.15 Fim_Se
3.3 Fim_Se
4. Senao
4.1 Se (FilaPostoB2Vez == []) Entao
    4.1.1 Se (FilaPostoB Tem Utentes Prioritarios) Entao
        4.1.1.1 Selecionar o utente prioritario da lista com
            menor indice
    4.1.2 Senao
        4.1.2.1 Selecionar o utente da frente da fila
    4.1.3 Fim_Se
    4.1.4 TempoChegadaUtente <- consultar valor na lista da
        FilaPostoB
    4.1.5 NumeroUtentesPostoB <- NumeroUtentesPostoB + 1
    4.1.6 TempoEsperaUtente <- Clock - TempoChegadaUtente
    4.1.7 TempoAtendimentoPostoB1 <- consultar o valor na
        lista da FilaPostoB
    4.1.8 TempoPartidaPostoB1 <- Clock +
        TempoAtendimentoPostoB1
    4.1.9 FilaPostoB <- RemoverUtenteFilaPostoB(idUtente)
    4.1.10 TempoTotalEsperaFilaPostoB <-
        TempoTotalEsperaFilaPostoB + TempoEsperaUtente
    4.1.11 TempoTotalOcupacaoPostoB1 <-
        TempoTotalOcupacaoPostoB1 + TempoAtendimentoPostoB1
    4.1.12 Se (Clock >= 0 E Clock <= 7200) Entao
        4.1.12.1 TempoTotalOcupacaoPostoB10911 <-
            TempoTotalOcupacaoPostoB10911 +
            TempoAtendimentoPostoB1
    4.1.13 Senao_Se (Clock > 7200 E Clock <= 14400) Entao
        4.1.13.1 TempoTotalOcupacaoPostoB11113 <-
            TempoTotalOcupacaoPostoB11113 +
            TempoAtendimentoPostoB1
    4.1.14 Senao_Se (Clock > 14400 E Clock <= 21600) Entao
        4.1.14.1 TempoTotalOcupacaoPostoB11315 <-
            TempoTotalOcupacaoPostoB11315 +
            TempoAtendimentoPostoB1
    4.1.15 Senao
        4.1.15.1 TempoTotalOcupacaoPostoB11517 <-
            TempoTotalOcupacaoPostoB11517 +
            TempoAtendimentoPostoB1
    4.1.16 Fim_Se

```



```
4.2 Senao
  4.2.1 Se (FilaPostoB Tem Utentes Prioritarios) Entao
    4.2.1.1 Selecionar o utente prioritario com menor indice
    4.2.1.2 Voltar ao ponto 4.1.4
  4.2.2 Senao
    4.2.2.1 Voltar ao ponto 3.2.1
  4.2.3 Fim_Se
4.3 Fim_Se
5. Fim_Se
6. Se (TempoAtendimentoTesouraria(Utente3Fase) != 0) Entao
  6.1 EventoChegadaTesouraria(indice , clock)
7. Fim_Se
```

Listing 3.13: Algoritmo do Evento de Partida do Posto B1

3.4.10 Evento de Partida do Posto B2

O algoritmo presente em 3.14 representa o Evento de Partida do Posto B2. Este evento é idêntico ao referido anteriormente, pelo qual o utente pode ter dois caminhos diferentes:

- O utente parte para a 3.^a fase, a Tesouraria;
- O utente abandona o sistema;

Neste caso, se não existirem utentes na fila de espera para o Posto B (FilaPostoB=[]), então este posto fica livre e o tempo da partida é alterado para indefinido/infinito. No caso de existirem utentes à espera para o balcão B, verifica-se se na fila existem utentes prioritários, no caso da sua presença, estes terão prioridade como o próprio nome indica, em relação aos clientes gerais. Depois de seleccionado o utente, prossegue-se ao cálculo do tempo de espera, do tempo da próxima partida e das restantes variáveis estatísticas.

```
Evento de Partida Posto B2 (clock):
1. Utente3Fase(indice) <- consultar valor na lista
   UtentesPostoB2
2. Eliminar o utente da lista dos UtentesPostoB2
3. Se (FilaPostoB = []) Entao
3.1 Se (FilaPostoB2Vez == []) Entao
3.1.1 EstadoPostoB2 <- livre
3.2.1 TempoPartidaPostoB2 <- INFINITO
3.2 Senao
3.2.1 Se (FilaPostoB2Vez Tem Utentes Prioritarios) Entao
3.2.1.1 Selecionar o utente prioritario da lista com
      menor indice
3.2.2 Senao
3.2.2.1 Selecionar o utente da frente da fila
3.2.3 Fim_Se
3.2.4 TempoChegadaUtente <- consultar valor na lista da
      FilaPostoB2Vez
3.2.5 TempoEsperaUtente <- Clock - TempoChegadaUtente
3.2.6 TempoAtendimentoPostoB2 <- consultar o valor na
      lista FilaPostoB2Vez
3.2.7 TempoPartidaPostoB2 <- Clock +
      TempoAtendimentoPostoB2
3.2.8 FilaPostoB2Vez <- RemoverUtenteFilaPostoB2Vez(
      idUtente)
3.2.9 TempoTotalEsperaFilaPostoB <-
      TempoTotalEsperaFilaPostoB + TempoEsperaUtente
3.2.10 TempoTotalOcupacaoPostoB2 <-
      TempoTotalOcupacaoPostoB2 + TempoAtendimentoPostoB2
3.2.11 Se (Clock >= 0 E Clock <= 7200) Entao
```

```

3.2.11.1 TempoTotalOcupacaoPostoB20911 <-
    TempoTotalOcupacaoPostoB20911 +
    TempoAtendimentoPostoB2
3.2.12 Senao_Se (Clock > 7200 E Clock <= 14400) Entao
    3.2.12.1 TempoTotalOcupacaoPostoB21113 <-
        TempoTotalOcupacaoPostoB21113 +
        TempoAtendimentoPostoB2
3.2.13 Senao_Se (Clock > 14400 E Clock <= 21600) Entao
    3.2.13.1 TempoTotalOcupacaoPostoB21315 <-
        TempoTotalOcupacaoPostoB21315 +
        TempoAtendimentoPostoB2
3.2.14 Senao
    3.2.14.1 TempoTotalOcupacaoPostoB21517 <-
        TempoTotalOcupacaoPostoB21517 +
        TempoAtendimentoPostoB2
3.2.15 Fim_Se
3.3 Fim_Se
4. Senao
4.1 Se (FilaPostoB2Vez == []) Entao
    4.1.1 Se (FilaPostoB Tem Utentes Prioritarios) Entao
        4.1.1.1 Selecionar o utente prioritario da lista com
            menor indice
    4.1.2 Senao
        4.1.2.1 Selecionar o utente da frente da fila
    4.1.3 Fim_Se
    4.1.4 TempoChegadaUtente <- consultar valor na lista da
        FilaPostoB
    4.1.5 NumeroUtentesPostoB <- NumeroUtentesPostoB + 1
    4.1.6 TempoEsperaUtente <- Clock - TempoChegadaUtente
    4.1.7 TempoAtendimentoPostoB2 <- consultar o valor na
        lista
    4.1.8 TempoPartidaPostoB2 <- Clock +
        TempoAtendimentoPostoB2
    4.1.9 FilaPostoB <- RemoverUtenteFilaPostoB(idUtente)
    4.1.10 TempoTotalEsperaFilaPostoB <-
        TempoTotalEsperaFilaPostoB + TempoEsperaUtente
    4.1.11 TempoTotalOcupacaoPostoB2 <-
        TempoTotalOcupacaoPostoB2 + TempoAtendimentoPostoB2
    4.1.12 Se (Clock >= 0 E Clock <= 7200) Entao
        4.1.12.1 TempoTotalOcupacaoPostoB20911 <-
            TempoTotalOcupacaoPostoB20911 +
            TempoAtendimentoPostoB2
    4.1.13 Senao_Se (Clock > 7200 E Clock <= 14400) Entao
        4.1.13.1 TempoTotalOcupacaoPostoB21113 <-
            TempoTotalOcupacaoPostoB21113 +
            TempoAtendimentoPostoB2
    4.1.14 Senao_Se (Clock > 14400 E Clock <= 21600) Entao
        4.1.14.1 TempoTotalOcupacaoPostoB21315 <-
            TempoTotalOcupacaoPostoB21315 +

```

```
TempoAtendimentoPostoB2
4.1.15 Senao
    4.1.15.1 TempoTotalOcupacaoPostoB21517 <-
        TempoTotalOcupacaoPostoB21517 +
        TempoAtendimentoPostoB2
4.1.16 Fim_Se
4.2 Senao
    4.2.1 Se (FilaPostoB Tem Utentes Prioritarios) Entao
        4.2.1.1 Selecionar o utente prioritario com menor indice
        4.2.1.2 Voltar ao ponto 4.1.4
    4.2.2 Senao
        4.2.2.1 Voltar ao ponto 3.2.1
    4.2.3 Fim_Se
4.3 Fim_Se
5. Fim_Se
6. Se (TempoAtendimentoTesouraria(Utente3Fase) != 0) Entao
    6.1 EventoChegadaTesouraria(indice , clock)
7. Fim_Se
```

Listing 3.14: Algoritmo do Evento de Partida do Posto B2

3.4.11 Evento de Chegada ao Posto C

Quando um cliente abandona o balcão da triagem, pode-se dirigir para os balcões A, B ou C (2.^a fase) ou dirigir-se para a tesouraria (3.^a fase) e seguidamente abandonar o sistema. O algoritmo apresentado em 3.15 explica o evento de Chegada ao Posto C. Quando um utente abandona a 2.^a fase e vai para a Tesouraria ainda pode voltar para trás, para a 2.^a fase, deve-se ter em atenção esta situação no algoritmo.

A variável presente no cabeçalho **nvez** informa-nos se o cliente está a chegar pela primeira vez (**nvez=1**) ou se pela segunda vez (**nvez=2**). Este balcão é idêntico ao da Chegada à Triagem. O que diferencia a chegada de um utente pela primeira ou segunda vez é que na segunda vez tem prioridade sobre os utentes que estão na fila do Posto C com excepção dos prioritários.

Deve-se verificar a disponibilidade do balcão C e os utentes vão saindo pouco a pouco à medida da disponibilidade do balcão.

```

Evento de Chegada Posto C (indiceUtente , clock , nvez):
1. Se (EstadoPostoC = ocupado) Entao
  1.1 Se (nvez = 2) Entao
    1.1.1 FilaPostoC2Vez <- InserirFilaPostoC2Vez(idUtente ,
      TempoChegada , tatendimento)
    1.1.2 TotalClientesFilaPostoC <- TotalClientesFilaPostoC +
      1
  1.2 Senao
    1.2.1 FilaPostoC <- InserirFilaPostoC(idUtente ,
      TempoChegada , tatendimento)
    1.2.2 TotalClientesFilaPostoC <- TotalClientesFilaPostoC +
      1
  1.3 Fim_Se
2. Senao
  2.1 Se (nvez = 2) Entao
    2.1.1 EstadoPostoC = ocupado
    2.1.2 TempoAtendimentoPostoC <- consultar o valor na
      listaUtentes(indiceUtente)
    2.1.3 TempoPartidaPostoC <- Clock + TempoAtendimentoPostoC
    2.1.4 TempoTotalOcupacaoPostoC <- TempoTotalOcupacaoPostoC
      + TempoAtendimentoPostoC
    2.1.5 Se (Clock >= 0 E Clock <= 7200) Ent o
      2.1.5.1 TempoTotalOcupacaoPostoC0911 <-
        TempoTotalOcupacaoPostoC0911 + TempoAtendimentoPostoC
    2.1.6 Sen oSe (Clock > 7200 E Clock <= 14400) Ent o
      2.1.6.1 TempoTotalOcupacaoPostoC1113 <-
        TempoTotalOcupacaoPostoC1113 + TempoAtendimentoPostoC
    2.1.7 Sen oSe (Clock > 14400 E Clock <= 21600) Ent o
      2.1.7.1 TempoTotalOcupacaoPostoC1315 <-
        TempoTotalOcupacaoPostoC1315 + TempoAtendimentoPostoC

```

```
2.1.8 Senao
    2.1.8.1 TempoTotalOcupacaoPostoC1517 <-
        TempoTotalOcupacaoPostoC1517 + TempoAtendimentoPostoC
2.1.9 Fim_Se
2.2 Senao
    2.2.1 EstadoPostoC = ocupado
    2.2.2 NumeroUtentesPostoC <- NumeroUtentesPostoC + 1
    2.2.3 TempoAtendimentoPostoC <- consultar o valor na
        listaUtentes(indiceUtente)
    2.2.4 TempoPartidaPostoC <- Clock + TempoAtendimentoPostoC
    2.2.5 TempoTotalOcupacaoPostoC <- TempoTotalOcupacaoPostoC
        + TempoAtendimentoPostoC
    2.2.6 Se (Clock >= 0 E Clock <= 7200) Entao
        2.2.6.1 TempoTotalOcupacaoPostoC0911 <-
            TempoTotalOcupacaoPostoC0911 + TempoAtendimentoPostoC
    2.2.7 Senao_Se (Clock > 7200 E Clock <= 14400) Entao
        2.2.7.1 TempoTotalOcupacaoPostoC1113 <-
            TempoTotalOcupacaoPostoC1113 + TempoAtendimentoPostoC
    2.2.8 Senao_Se (Clock > 14400 E Clock <= 21600) Entao
        2.2.8.1 TempoTotalOcupacaoPostoC1315 <-
            TempoTotalOcupacaoPostoC1315 + TempoAtendimentoPostoC
    2.2.9 Senao
        2.2.9.1 TempoTotalOcupacaoPostoC1517 <-
            TempoTotalOcupacaoPostoC1517 + TempoAtendimentoPostoC
    2.2.10 Fim_Se
2.3 Fim_Se
3. Fim_Se
```

Listing 3.15: Algoritmo do Evento de Chegada ao Posto C

3.4.12 Evento de Partida do Posto C

Este pseudocódigo representa o Evento de Partida do Posto C do sistema, que se encontra representado em 3.16. Depois de um cliente abandonar o balcão do Posto C, o cliente pode ter duas vias, nomeadamente:

- O utente vai para a 3.^a fase, a Tesouraria;
- O utente abandona o sistema.

Assim, caso não existam utentes na fila de espera do Posto C (FilaPostoC=[]), este posto fica livre e o tempo da próxima partida é alterado para infinito. Caso existam clientes na fila de espera, então verifica-se se há presença de utentes prioritários, se houver estes seguem primeiro consoante a ordem de chegada e depois os gerais. Depois de selecionado o utente devido, prossegue-se para o cálculo do tempo de espera, do tempo de partida e das variáveis estatísticas.

Evento de Partida Posto C (clock):

1. Ute3Fase(indice) ← consultar valor na lista UtePostoC
2. Eliminar o utente da lista dos UtePostoC
3. Se (FilaPostoC = []) Entao
 - 3.1 Se (FilaPostoC2Vez == []) Entao
 - 3.1.1 EstadoPostoC ← livre
 - 3.1.2 TempoPartidaPostoC ← INFINITO
 - 3.2 Senao
 - 3.2.1 Se (FilaPostoC2Vez Tem Ute Prioritarios) Entao
 - 3.2.1.1 Selecionar o utente prioritario da lista com menor indice
 - 3.2.2 Senao
 - 3.2.2.1 Selecionar o utente da frente da fila
 - 3.2.3 Fim_Se
 - 3.2.4 TempoChegadaUte ← consultar valor na lista da FilaPostoC
 - 3.2.5 TempoEsperaUte ← Clock – TempoChegadaUte
 - 3.2.6 TempoAtendimentoPostoC ← consultar o valor na lista da FilaPostoC
 - 3.2.7 TempoPartidaPostoC ← Clock + TempoAtendimentoPostoC
 - 3.2.8 FilaPostoC2Vez ← RemoverUteFilaPostoC2Vez(idUte)
 - 3.2.9 TempoTotalEsperaFilaPostoC ← TempoTotalEsperaFilaPostoC + TempoEsperaUte
 - 3.2.10 TempoTotalOcupacaoPostoC ← TempoTotalOcupacaoPostoC + TempoAtendimentoPostoC
 - 3.2.11 Se (Clock >= 0 E Clock <= 7200) Entao
 - 3.2.11.1 TempoTotalOcupacaoPostoC0911 ← TempoTotalOcupacaoPostoC0911 + TempoAtendimentoPostoC

```
3.2.12 Senao_Se (Clock > 7200 E Clock <= 14400) Entao
    3.2.12.1 TempoTotalOcupacaoPostoC1113 <-
        TempoTotalOcupacaoPostoC1113 + TempoAtendimentoPostoC
3.2.13 Senao_Se (Clock > 14400 E Clock <= 21600) Entao
    3.2.13.1 TempoTotalOcupacaoPostoC1315 <-
        TempoTotalOcupacaoPostoC1315 + TempoAtendimentoPostoC
3.2.14 Senao
    3.2.14.1 TempoTotalOcupacaoPostoC1517 <-
        TempoTotalOcupacaoPostoC1517 + TempoAtendimentoPostoC
3.2.15 Fim_Se
4. Senao
4.1 Se (FilaPostoC2Vez == []) Entao
    4.1.1 Se (FilaPostoC Tem Utentes Prioritarios) Entao
        4.1.1.1 Selecionar o utente prioritario da lista com
            menor indice
    4.1.2 Senao
        4.1.2.1 Selecionar o utente da frente da fila
    4.1.3 Fim_Se
    4.1.4 TempoChegadaUtente <- consultar valor na lista da
        FilaPostoC
    4.1.5 NumeroUtentesPostoC <- NumeroUtentesPostoC + 1
    4.1.6 TempoEsperaUtente <- Clock - TempoChegadaUtente
    4.1.7 TempoAtendimentoPostoC <- consultar o valor na lista
        da FilaPostoC
    4.1.8 TempoPartidaPostoC <- Clock + TempoAtendimentoPostoC
    4.1.9 FilaPostoC <- RemoverUtenteFilaPostoC(idUtente)
    4.1.10 TempoTotalEsperaFilaPostoC <-
        TempoTotalEsperaFilaPostoC + TempoEsperaUtente
    4.1.11 TempoTotalOcupacaoPostoC <-
        TempoTotalOcupacaoPostoC + TempoAtendimentoPostoC
    4.1.12 Se (Clock >= 0 E Clock <= 7200) Entao
        4.1.12.1 TempoTotalOcupacaoPostoC0911 <-
            TempoTotalOcupacaoPostoC0911 + TempoAtendimentoPostoC
    4.1.13 Senao_Se (Clock > 7200 E Clock <= 14400) Entao
        4.1.13.1 TempoTotalOcupacaoPostoC1113 <-
            TempoTotalOcupacaoPostoC1113 + TempoAtendimentoPostoC
    4.1.14 Senao_Se (Clock > 14400 E Clock <= 21600) Entao
        4.1.14.1 TempoTotalOcupacaoPostoC1315 <-
            TempoTotalOcupacaoPostoC1315 + TempoAtendimentoPostoC
    4.1.15 Senao
        4.1.15.1 TempoTotalOcupacaoPostoC1517 <-
            TempoTotalOcupacaoPostoC1517 + TempoAtendimentoPostoC
    4.1.16 Fim_Se
4.2 Senao
    4.2.1 Se (FilaPostoC Tem Utentes Prioritarios) Entao
        4.2.1.1 Selecionar o utentes prioritario com menor
            indice
        4.2.1.2 Voltar ao ponto 4.1.4
    4.2.2 Senao
```



```
4.2.2.1 Voltar ao ponto 3.2.1
4.2.3 Fim_Se
4.3 Fim_Se
5. Fim_Se
6. Se (TempoAtendimentoTesouraria(Utente3Fase) != 0) Entao
    6.1 EventoChegadaTesouraria(indice , clock)
7. Fim_Se
```

Listing 3.16: Algoritmo do Evento de Partida ao Posto C

3.4.13 Evento de Chegada à Tesouraria

No algoritmo representado em 3.17 apresenta o Evento de Chegada à Tesouraria. Um utente quando abandona o balcão da Triagem pode dirigir-se directamente para a 3.^a fase, a Tesouraria ou quando um utente abandona um dos balcões (A, B ou C) da 2.^a fase também se pode dirigir à Tesouraria. Neste algoritmo verifica-se se o balcão da Tesouraria está ocupado (visível no ponto 1). Caso este esteja ocupado, então o utente é colocado na fila de espera (FilaTesouraria), se não o utente é atendido no balcão e calcula-se o respectivo tempo de partida do balcão, bem como o valor das outras variáveis estatísticas nos períodos de tempo parciais de 2h e no tempo global de 8h.

```

Evento de Chegada Tesouraria (indiceUtente , clock):
1. Se (EstadoTesouraria = ocupado) Entao
  1.1 FilaTesouraria <- InserirFilaTesouraria(idUtente ,
    TempoChegada)
  1.2 TotalClientesFilaTesouraria <-
    TotalClientesFilaTesouraria + 1
2. Senao
  2.1 EstadoTesouraria = ocupado
  4.4 NumeroUtentesTesouraria <- NumeroUtentesTesouraria + 1
  2.2 TempoAtendimentoTesouraria <- consultar o valor na
    listaUtentes(indiceUtente)
  2.3 TempoPartidaTesouraria <- Clock +
    TempoAtendimentoTesouraria
  2.4 TempoTotalOcupacaoTesouraria <-
    TempoTotalOcupacaoTesouraria + TempoAtendimentoTesouraria
  2.5 Se (Clock >= 0 E Clock <= 7200) Entao
    2.5.1 TempoTotalOcupacaoTesouraria0911 <-
      TempoTotalOcupacaoTesouraria0911 +
      TempoAtendimentoTesouraria
  2.6 Senao_Se (Clock > 7200 E Clock <= 14400) Entao
    2.6.1 TempoTotalOcupacaoTesouraria1113 <-
      TempoTotalOcupacaoTesouraria1113 +
      TempoAtendimentoTesouraria
  2.7 Senao_Se (Clock > 14400 E Clock <= 21600) Entao
    2.7.1 TempoTotalOcupacaoTesouraria1315 <-
      TempoTotalOcupacaoTesouraria1315 +
      TempoAtendimentoTesouraria
  2.8 Senao
    2.8.1 TempoTotalOcupacaoTesouraria1517 <-
      TempoTotalOcupacaoTesouraria1517 +
      TempoAtendimentoTesouraria
  2.9 Fim_Se
3. Fim_Se

```

Listing 3.17: Evento de Chegada À Tesouraria

3.4.14 Evento de Partida da Tesouraria

O pseudocódigo presente em 3.18 está associado ao Evento de Partida da Tesouraria. Depois de um utente abandonar o balcão da Tesouraria, podem suceder-se dois acontecimentos:

- O utente abandona o sistema;
- O utente regressa à 2.^a fase, aos balcões A, B e C.

Desta forma caso não existam utentes na fila de espera da Tesouraria ($FilaTesouraria=[]$), então este posto fica livre e o tempo da próxima partida é alterado para infinito. Caso existam clientes na fila de espera, então verifica-se se existem clientes prioritários, se existirem, estes terão prioridade sobre os restantes. Depois de seleccionado o utente, prossegue-se ao cálculo do tempo de espera, do tempo da próxima partida e das restantes variáveis estatísticas.

Evento de Partida Tesouraria (clock):

1. $Utente2Fase2(indice) \leftarrow$ consultar valor na lista $UtentesTesouraria$
2. Eliminar o utente da lista dos $UtentesTesouraria$
3. Se ($FilaTesouraria = []$) Entao
 - 3.1 $EstadoTesouraria \leftarrow$ livre
 - 3.2 $TempoPartidaTesouraria \leftarrow$ INFINITO
4. Senao
 - 4.1 Se ($FilaTesouraria$ Tem Utesntes Prioritarios) Entao
 - 4.1.1 Selecionar o utente prioritario da lista com menor indice
 - 4.2 Senao
 - 4.2.1 Selecionar o utente da frente da fila
 - 4.3 $TempoChegadaUtente \leftarrow$ consultar valor na lista da $FilaTesouraria$
 - 4.4 $NumeroUtentesTesouraria \leftarrow$ $NumeroUtentesTesouraria + 1$
 - 4.5 $TempoEsperaUtente \leftarrow$ $Clock - TempoChegadaUtente$
 - 4.6 $TempoAtendimentoTesouraria \leftarrow$ consultar o valor na lista da $FilaTesouraria$
 - 4.7 $TempoPartidaTesouraria \leftarrow$ $Clock +$
 $TempoAtendimentoTesouraria$
 - 4.8 $FilaTesouraria \leftarrow$ $RemoverUtenteFilaTesouraria(idUtente)$
 - 4.9 $TempoTotalEsperaFilaTesouraria \leftarrow$
 $TempoTotalEsperaFilaTesouraria + TempoEsperaUtente$
 - 4.10 $TempoTotalOcupacaoTesouraria \leftarrow$
 $TempoTotalOcupacaoTesouraria + TempoAtendimentoTesouraria$
 - 4.11 Se ($Clock \geq 0$ E $Clock \leq 7200$) Entao
 - 4.11.1 $TempoTotalOcupacaoTesouraria0911 \leftarrow$
 $TempoTotalOcupacaoTesouraria0911 +$
 $TempoAtendimentoTesouraria$

```
4.12 Senao_Se (Clock > 7200 E Clock <= 14400) Entao
    4.12.1 TempoTotalOcupacaoTesouraria1113 <-
        TempoTotalOcupacaoTesouraria1113 +
        TempoAtendimentoTesouraria
4.13 Senao_Se (Clock > 14400 E Clock <= 21600) Entao
    4.13.1 TempoTotalOcupacaoTesouraria1315 <-
        TempoTotalOcupacaoTesouraria1315 +
        TempoAtendimentoTesouraria
4.14 Senao
    4.14.1 TempoTotalOcupacaoTesouraria1517 <-
        TempoTotalOcupacaoTesouraria1517 +
        TempoAtendimentoTesouraria
4.15 Fim_Se
5. Fim_Se
6. Se (TempoAtendimento2Fase2(Utente2Fase2) != 0 ) Entao
    6.1 TipoAssunto <- consultar valor na ListaUtentes(
        Utente2Fase2)
    6.2. Se (TipoAssunto <- A) Entao
        6.2.1 EventoChegadaPostoA(indice ,clock ,2)
    6.3 Senao_Se (TipoAssunto <- B) Entao
        6.3.1 EventoChegadaPostoB(indice ,clock ,2)
    6.4 Senao_Se (TipoAssunto <- C) Entao
        6.4.1 EventoChegadaPostoC(indice ,clock ,2)
    6.5 Fim_Se
7. Fim_Se
```

Listing 3.18: Evento de Partida da Tesouraria

3.4.15 Programa Principal

O pseudocódigo presente em 3.19 representa o desenvolvimento do Programa Principal. Neste programa é possível verificar que são inicializadas todas as variáveis estatísticas (tempo da próxima chegada, tempo da próxima triagem, tempo partida do Posto A1 e A2, tempo partida do Posto B1 e B2, tempo partida do Posto C e tempo da próxima partida da Tesouraria). Se existe uma chegada ao sistema, o número de utentes presentes no sistema é incrementado para +1. Quando o número de utentes no sistema é igual ao total (MaximoUtentes) então o tempo para a próxima chegada modifica-se para infinito, verificando-se sempre os eventos de partida referidos anteriormente. Por fim realiza-se o cálculo das medidas de desempenho e dá-se o fim da simulação do sistema.

Programa Principal:

1. Inicializar todas as variaveis
2. TChegada \leftarrow consultar listUtentes
3. (Clock, TipoEvento) \leftarrow rotina de gestao de tempo (TPChegada, TPTriagem, TPPostoA1, TPPostoA2, TPPostoB1, TPPostoB2, TPPostoC, TPPTesouraria)
4. Se (TipoEvento = Chegada) Entao
 - 4.1 NUtentesSistema = NUtentesSistema + 1
 - 4.2 Executar rotina eventoChegada(Clock)
 - 4.3 Se (NUtentesSistema = MaximoUtentes) Entao
 - 4.3.1 TPChegada = INFINITO
 - 4.4 Fim_Se
5. Senao_Se (TipoEvento = Partida Triagem) Entao
 - 5.1 Executar rotina eventoPartidaTriagem(Clock)
6. Senao_Se (TipoEvento = Partida PostoA1) Entao
 - 6.1 Executar rotina eventoPartidaPostoA1(Clock)
7. Senao_Se (TipoEvento = Partida PostoA2) Entao
 - 7.1 Executar rotina eventoPartidaPostoA2(Clock)
8. Senao_Se (TipoEvento = Partida PostoB1) Entao
 - 8.1 Executar rotina eventoPartidaPostoB1(Clock)
9. Senao_Se (TipoEvento = Partida PostoB2) Entao
 - 9.1 Executar rotina eventoPartidaPostoB2(Clock)
10. Senao_Se (TipoEvento = Partida PostoC) Entao
 - 10.1 Executar rotina eventoPartidaPostoC(Clock)
11. Senao_Se (TipoEvento = Partida Tesouraria) Entao
 - 11.1 Executar rotina eventoPartidaTesouraria(Clock)
12. Senao
 - 12.1 Terminar simulacao \leftarrow True
13. Fim_Se
14. Se (Continuar simulacao) Entao
 - 14.1 Regressar a 3.
15. Senao
 - 15.1 Calcular medidas de desempenho

```
15.2 Escrever Relatorio  
16. Fim_Se
```

Listing 3.19: Programa Principal

3.5 Conclusão

Neste capítulo foram discutidos todos os tipos de eventos criados para

Capítulo 4

Análise de Resultados

De forma a analisar o sistema de simulação desenvolvido, foi feita uma centena de simulações e registados os valores das variáveis estatísticas num ficheiro de texto. Procedeu-se ao cálculo da média dos valores obtidos e os resultados são os ilustrados na Tabela 4.1.

Na coluna **Resultado**, no caso das variáveis temporais, é apresentado o tempo em segundos e entre parênteses a conversão para horas:minutos:segundos.

Através da análise da tabela, pode concluir-se que o Posto B é o que mais utentes tem durante o dia. O tempo de espera médio na fila ronda os 4 minutos. A fila da Tesouraria é a fila onde o tempo de espera médio é menor, sempre abaixo dos 5 segundos, um tempo quase imperceptível na realidade.

Tabela 4.1: Resultados médios finais das variáveis estatísticas do sistema

Variável	Descrição	Resultado
<i>NUSistema</i>	N.º de utentes no sistema	135
<i>TUFTriagem</i>	Total de utentes na fila da Triagem	47
<i>TTEsperaTriagem</i>	Tempo médio de espera na fila da triagem	30 (0:00:30)
<i>TTEsperaTriagem0911</i>	Tempo médio de espera na fila da triagem (09h00-11h00)	1 (0:00:01)
<i>TTEsperaTriagem1113</i>	Tempo médio de espera na fila da triagem (11h00-13h00)	4 (0:00:04)
<i>TTEsperaTriagem1315</i>	Tempo médio de espera na fila da triagem (13h00-15h00)	22 (0:00:22)
<i>TTEsperaTriagem1517</i>	Tempo médio de espera na fila da triagem (15h00-17h00)	4 (0:00:04)
<i>TTOcupacaoTriagem</i>	Tempo médio de ocupação do balcão da Triagem	8416 (2:20:16)
<i>TTOcupacaoTriagem0911</i>	Tempo médio de ocupação do balcão da Triagem (09h00-11h00)	940 (0:15:40)
<i>TTOcupacaoTriagem1113</i>	Tempo médio de ocupação do balcão da Triagem (11h00-13h00)	2000 (0:33:20)
<i>TTOcupacaoTriagem1315</i>	Tempo médio de ocupação do balcão da Triagem (13h00-15h00)	3625 (1:00:25)
<i>TTOcupacaoTriagem1517</i>	Tempo médio de ocupação do balcão da Triagem (15h00-17h00)	1850 (0:30:50)
<i>NUPostoA</i>	N.º de utentes no Posto A	45
<i>TUFPostoA</i>	Total de utentes na fila do Posto A	33
<i>TTEsperaPostoA</i>	Tempo médio de espera na fila do Posto A	1140 (0:19:00)
<i>TTEsperaPostoA0911</i>	Tempo médio de espera na fila do Posto A (09h00-11h00)	9 (0:00:09)
<i>TTEsperaPostoA1113</i>	Tempo médio de espera na fila do Posto A (11h00-13h00)	60 (0:01:00)
<i>TTEsperaPostoA1315</i>	Tempo médio de espera na fila do Posto A (13h00-15h00)	377 (0:06:17)
<i>TTEsperaPostoA1517</i>	Tempo médio de espera na fila do Posto A (15h00-17h00)	695 (0:11:35)
<i>TTOcupacaoPostoA1</i>	Tempo médio de ocupação do Posto A1	20699 (5:44:59)
<i>TTOcupacaoPostoA10911</i>	Tempo médio de ocupação do Posto A1 (09h00-11h00)	2929 (0:48:49)
<i>TTOcupacaoPostoA11113</i>	Tempo médio de ocupação do Posto A1 (11h00-13h00)	4891 (1:21:31)
<i>TTOcupacaoPostoA11315</i>	Tempo médio de ocupação do Posto A1 (13h00-15h00)	6906 (1:55:06)
<i>TTOcupacaoPostoA11517</i>	Tempo médio de ocupação do Posto A1 (15h00-17h00)	5974 (1:39:34)
<i>TTOcupacaoPostoA2</i>	Tempo médio de ocupação do Posto A2	17637 (4:53:57)
<i>TTOcupacaoPostoA20911</i>	Tempo médio de ocupação do Posto A2 (09h00-11h00)	1224 (0:20:24)
<i>TTOcupacaoPostoA21113</i>	Tempo médio de ocupação do Posto A2 (11h00-13h00)	3961 (1:06:01)
<i>TTOcupacaoPostoA21315</i>	Tempo médio de ocupação do Posto A2 (13h00-15h00)	6984 (1:56:24)
<i>TTOcupacaoPostoA21517</i>	Tempo médio de ocupação do Posto A2 (15h00-17h00)	5467 (1:31:07)
<i>NUPostoB</i>	N.º de utentes do Posto B	60
<i>TUFPostoB</i>	Total de utentes na fila do Posto B	32
<i>TTEsperaPostoB</i>	Tempo médio de espera na fila do Posto B	293 (0:04:53)

Variável	Descrição	Resultado
<i>TTEsperaPostoB0911</i>	Tempo médio de espera na fila do Posto B (09h00-11h00)	1 (0:00:01)
<i>TTEsperaPostoB1113</i>	Tempo médio de espera na fila do Posto B (11h00-15h00)	59 (0:00:59)
<i>TTEsperaPostoB1315</i>	Tempo médio de espera na fila do Posto B (13h00-15h00)	150 (0:02:30)
<i>TTEsperaPostoB1517</i>	Tempo médio de espera na fila do Posto B (15h00-17h00)	83 (0:01:23)
<i>TTOcupacaoPostoB1</i>	Tempo médio de ocupação do Posto B1	19207 (5:20:07)
<i>TTOcupacaoPostoB10911</i>	Tempo médio de ocupação do Posto B1 (09h00-11h00)	2855 (0:47:35)
<i>TTOcupacaoPostoB11113</i>	Tempo médio de ocupação do Posto B1 (11h00-13h00)	4959 (1:22:39)
<i>TTOcupacaoPostoB11315</i>	Tempo médio de ocupação do Posto B1 (13h00-15h00)	6355 (1:45:55)
<i>TTOcupacaoPostoB11517</i>	Tempo médio de ocupação do Posto B1 (15h00-17h00)	5038 (1:23:58)
<i>TTOcupacaoPostoB2</i>	Tempo médio de ocupação do Posto B2	14681 (4:04:41)
<i>TTOcupacaoPostoB20911</i>	Tempo médio de ocupação do Posto B2 (09h00-11h00)	1000 (0:16:40)
<i>TTOcupacaoPostoB21113</i>	Tempo médio de ocupação do Posto B2 (11h00-13h00)	4216 (1:10:16)
<i>TTOcupacaoPostoB21315</i>	Tempo médio de ocupação do Posto B2 (13h00-15h00)	6076 (1:41:16)
<i>TTOcupacaoPostoB21517</i>	Tempo médio de ocupação do Posto B2 (15h00-17h00)	3389 (0:56:29)
<i>NUPostoC</i>	N.º de utentes no Posto C	16
<i>TUFPostoC</i>	Total de utentes na fila do Posto C	12
<i>TTEsperaPostoC</i>	Tempo médio de espera na fila do Posto C	693 (0:11:33)
<i>TTEsperaPostoC0911</i>	Tempo médio de espera na fila do Posto C (09h00-11h00)	10 (0:00:10)
<i>TTEsperaPostoC1113</i>	Tempo médio de espera na fila do Posto C (11h00-13h00)	75 (0:01:15)
<i>TTEsperaPostoC1315</i>	Tempo médio de espera na fila do Posto C (13h00-15h00)	339 (0:05:39)
<i>TTEsperaPostoC1517</i>	Tempo médio de espera na fila do Posto C (15h00-17h00)	270 (0:04:30)
<i>TTOcupacaoPostoC</i>	Tempo médio de ocupação do Posto C	14730 (4:05:30)
<i>TTOcupacaoPostoC0911</i>	Tempo médio de ocupação do Posto C (09h00-11h00)	1211 (0:20:11)
<i>TTOcupacaoPostoC1113</i>	Tempo médio de ocupação do Posto C (11h00-13h00)	3290 (0:54:50)
<i>TTOcupacaoPostoC1315</i>	Tempo médio de ocupação do Posto C (13h00-15h00)	5652 (1:34:12)
<i>TTOcupacaoPostoC1517</i>	Tempo médio de ocupação do Posto C (15h00-17h00)	4576 (1:16:16)

Variável	Descrição	Resultado
<i>NU</i> Tesouraria	N.º de utentes na Tesouraria	52
<i>TU</i> FTesouraria	Total de utentes na fila da Tesouraria	8
<i>TT</i> EsperaTesouraria	Tempo médio de espera na fila da Tesouraria	8 (0:00:08)
<i>TT</i> EsperaTesouraria0911	Tempo médio de espera na fila da Tesouraria (09h00-11h00)	0 (0:00:00)
<i>TT</i> EsperaTesouraria1113	Tempo médio de espera na fila da Tesouraria (11h00-13h00)	2 (0:00:02)
<i>TT</i> EsperaTesouraria1315	Tempo médio de espera na fila da Tesouraria (13h00-15h00)	3 (0:00:03)
<i>TT</i> EsperaTesouraria1517	Tempo médio de espera na fila da Tesouraria (15h00-17h00)	2 (0:00:02)
<i>TT</i> OcupacaoTesouraria	Tempo médio de ocupação da Tesouraria	4080 (1:08:00)
<i>TT</i> OcupacaoTesouraria0911	Tempo médio de ocupação da Tesouraria (09h00-11h00)	443 (0:07:23)
<i>TT</i> OcupacaoTesouraria1113	Tempo médio de ocupação da Tesouraria (11h00-13h00)	1059 (0:17:39)
<i>TT</i> OcupacaoTesouraria1315	Tempo médio de ocupação da Tesouraria (13h00-15h00)	1498 (0:24:58)
<i>TT</i> OcupacaoTesouraria1517	Tempo médio de ocupação da Tesouraria (15h00-17h00)	1081 (0:18:01)

Capítulo 5

Conclusão

Em suma, o objectivo do trabalho foi realizar um modelo de simulação de um sistema relacionado com o funcionamento do Serviço de Atendimento de uma repartição de Finanças.

Verificámos que a execução de modelos de simulação realizada com recurso a um sistema computacional, tem um enorme potencial ao fornecer resultados muito precisos, sem ser necessário interferir num sistema real de Finanças. Os resultados analisados podem auxiliar as entidades competentes para a tomada de decisões que visam a resolução de problemas existentes nas Finanças no dia-a-dia.

Capítulo 6

Anexos

6.1 Geração dos Tempos para os Utentes

```
import random
import datetime

# OBJETIVO: Pretende-se construir e implementar um modelo de
#          simulação de um sistema relacionado
# com o funcionamento do Serviço de Atendimento de uma
# Repartição de Finanças, para averiguar o
# desempenho da solução apresentada.

##### SECCAO DAS FUNCOES #####

# OBJETIVO: gerar o número de clientes (entre 120 e 150)
# RETURN: número de clientes (nCliente)
def getNumeroClientes():
    return random.randrange(120,150)

# número de clientes
nClientes = getNumeroClientes()

# tabela com a informação de todos os clientes
tabelaClientes = []
```

```
# OBJETIVO: função que gera n meros entre 0 e 1 para um dado
#            n. de clientes
# PARAMETROS: seed -> Seed value is the previous value number
#              generated by the generator. (Ajustar p/ as %)
#              nClientes -> n mero de clientes
# RETURN: lista com os nClientes n meros aleatórios entre 0 e
#         1
def getListNumerosEntre0e1(seed, nClientes):
    listNum01 = []

    # definir o valor de seed
    random.seed(seed)
    for x in range(0, nClientes):
        # gerar um numero aleatorio entre 0 e 1
        num = random.random()
        listNum01.append(num)

    return listNum01

# OBJETIVO: função que gera os clientes considerados
#           prioritários
# PARAMETROS: listNumbers -> lista de n meros aleatórios entre
#           0 e 1
#              listClientes -> lista de clientes
# RETURN: lista com os clientes considerados prioritários
def getClientesPrioritarios(listNumbers, listClientes):
    listPrioritarios = []
    bkList = list(listClientes)

    for num in listNumbers:
        # 20% dos utentes são considerados prioritários logo
        # na 1ª fase, mantendo esta propriedade até partirem
        # (abandonarem) do sistema
        if num >= 0 and num <= 0.2:
            cliente = random.choice(list(bkList))
            listPrioritarios.append(cliente)
            bkList.remove(cliente)

    return listPrioritarios
```

```
# OBJETIVO: fun    o que gera os tempos de chegada
# PARAMETROS: listNumbers -> lista de n meros aleat rios entre
    0 e 1
# RETURN: lista com os tempos de chegada para os nClientes
def getTemposChegada(listNumbers):
    # lista c/ tempos das 9h -> 11h
    listInt1 = []
    # lista c/ tempos das 11h -> 13h
    listInt2 = []
    # lista c/ tempos das 13h -> 15h
    listInt3 = []
    # lista c/ tempos das 15h -> 17h
    listInt4 = []

    for num in listNumbers:
        if num >= 0 and num <= 0.1:
            # gerar um numero entre 0 e 7200
            listInt1.append(random.randrange(0,7200))
        elif num >= 0.11 and num <= 0.35:
            # gerar um numero entre 7201 e 14400
            listInt2.append(random.randrange(7201,14400))
        elif num >= 0.36 and num <= 0.80:
            # gerar um numero entre 14401 e 21600
            listInt3.append(random.randrange(14401,21600))
        else:
            # gerar um numero entre 21601 e 28800
            listInt4.append(random.randrange(21601,28800))

    return listInt1 + listInt2 + listInt3 + listInt4

# OBJETIVO: fun    o que gera os tempos de triagem
# PARAMETROS: listNumbers -> lista de n meros aleat rios entre
    0 e 1
# RETURN: lista com os tempos de triagem para os nClientes
def getTemposTriagem(listNumbers):
    listTemposTriagem = []

    for num in listNumbers:
        if num >= 0 and num <= 0.55:
            listTemposTriagem.append(random.randrange(1,60))
        elif num >= 0.56 and num <= 0.90:
            listTemposTriagem.append(random.randrange(60,120))
        else:
            listTemposTriagem.append(random.randrange(120,180))

    return listTemposTriagem
```

```
# OBJETIVO: fun    o que gera os clientes que passam da 1.
               para 3.    fase
# PARAMETROS: listNumbers -> lista de n meros aleat rios entre
               0 e 1
#               listClientes -> lista de clientes
# RETURN: lista com os tempos de chegada dos clientes que passam
               da 1.    para a 2.    fase
def getCientes1para3Fase(listNumbers , listClientes):
    listClientes1para3Fase = []
    bkList = list(listClientes)

    for num in listNumbers:
        # 10% dos utentes passam da 1    fase diretamente para a
            3    fase
        if num >= 0 and num <= 0.1:
            cliente = random.choice(list(bkList))
            listClientes1para3Fase.append(cliente)
            bkList.remove(cliente)

    return listClientes1para3Fase

# OBJETIVO: fun    o que gera os clientes que s o atendidos no
               Posto A
# PARAMETROS: listNumbers -> lista de n meros aleat rios entre
               0 e 1
#               listCli2Fase -> lista de clientes
# RETURN: lista com os clientes que s o atendidos no Posto A
def getCientesPostoA(listNumbers , listCli2Fase):
    listaClientesPostoA = []
    bkList = list(listCli2Fase)

    for num in listNumbers:
        # 35% dos utentes que passam pela 2    fase s o
            atendidos nos postos do tipo A
        if num >= 0 and num <= 0.35:
            cliente = random.choice(list(bkList))
            listaClientesPostoA.append(cliente)
            bkList.remove(cliente)

    return listaClientesPostoA
```

```
# OBJETIVO: função que gera os clientes que são atendidos no
# Posto C
# PARAMETROS: listNumbers -> lista de números aleatórios entre
# 0 e 1
# listClientes -> lista de clientes
# RETURN: lista com os clientes que são atendidos no Posto C
def getClientesPostoC(listNumbers, listClientes):
    listaClientesPostoC = []
    bkList = list(listClientes)

    for num in listNumbers:
        # 15% dos utentes que passam pela 2.ª fase são
        # atendidos nos postos do tipo C
        if num >= 0 and num <= 0.15:
            cliente = random.choice(list(bkList))
            listaClientesPostoC.append(cliente)
            bkList.remove(cliente)

    return listaClientesPostoC

# OBJETIVO: função que gera os clientes que são atendidos no
# Posto A e passam pela 3.ª fase
# PARAMETROS: listNumbers -> lista de números aleatórios entre
# 0 e 1
# listClientes -> lista de clientes
# RETURN: lista com os clientes que são atendidos no Posto A e
# passam pela 3.ª fase
def getClientesPostoA3Fase(listNumbers, listClientes):
    listaClientesPostoA3Fase = []
    bkList = list(listClientes)

    for num in listNumbers:
        # 20% dos utentes atendidos nos postos do tipo A passam
        # pela 3.ª fase
        if num >= 0 and num <= 0.2:
            cliente = random.choice(list(bkList))
            listaClientesPostoA3Fase.append(cliente)
            bkList.remove(cliente)

    return listaClientesPostoA3Fase
```



```
# OBJETIVO: função que gera os clientes que são atendidos no
# Posto A e passam pela 3. fase e regressam 2. fase
# PARAMETROS: listNumbers -> lista de números aleatórios entre
# 0 e 1
# listClientes -> lista de clientes
# RETURN: lista com os clientes que são atendidos no Posto A e
# passam pela 3. fase e regressam 2. fase
def getClientesPostoA3FaseRegressam(listNumbers, listClientes):
    listaClientes = []
    bkList = list(listClientes)

    for num in listNumbers:
        # apenas 30% regressam 2 fase
        if num >= 0 and num <= 0.3:
            cliente = random.choice(list(bkList))
            listaClientes.append(cliente)
            bkList.remove(cliente)

    return listaClientes

# OBJETIVO: função que gera os clientes que são atendidos no
# Posto B e passam pela 3. fase
# PARAMETROS: listNumbers -> lista de números aleatórios entre
# 0 e 1
# listClientes -> lista de clientes
# RETURN: lista com os clientes que são atendidos no Posto B e
# passam pela 3. fase
def getClientesPostoB3Fase(listNumbers, listClientes):
    listaClientes = []
    bkList = list(listClientes)

    for num in listNumbers:
        # 30% dos utentes atendidos nos postos do tipo B passam
        # pela 3 fase
        if num >= 0 and num <= 0.3:
            cliente = random.choice(list(bkList))
            listaClientes.append(cliente)
            bkList.remove(cliente)

    return listaClientes
```

```
# OBJETIVO: função que gera os clientes que são atendidos no
# Posto B e passam pela 3. fase e regressam 2. fase
# PARAMETROS: listNumbers -> lista de números aleatórios entre
# 0 e 1
# listClientes -> lista de clientes
# RETURN: lista com os clientes que são atendidos no Posto B e
# passam pela 3. fase e regressam 2. fase
def getClientesPostoB3FaseRegressam(listNumbers, listClientes):
    listaClientes = []
    bkList = list(listClientes)

    for num in listNumbers:
        # 20% regressam 2 fase
        if num >= 0 and num <= 0.2:
            cliente = random.choice(list(bkList))
            listaClientes.append(cliente)
            bkList.remove(cliente)

    return listaClientes

# OBJETIVO: função que gera os clientes que são atendidos no
# Posto C e passam pela 3. fase
# PARAMETROS: listNumbers -> lista de números aleatórios entre
# 0 e 1
# listClientes -> lista de clientes
# RETURN: lista com os clientes que são atendidos no Posto C e
# passam pela 3. fase
def getClientesPostoC3Fase(listNumbers, listClientes):
    listaClientes = []
    bkList = list(listClientes)

    for num in listNumbers:
        # 75% dos utentes atendidos nos postos do tipo C passam
        # pela 3 fase
        if num >= 0 and num <= 0.75:
            cliente = random.choice(list(bkList))
            listaClientes.append(cliente)
            bkList.remove(cliente)

    return listaClientes
```

```

# OBJETIVO: função que gera os clientes que são atendidos no
# Posto C e passam pela 3. fase e regressam 2. fase
# PARAMETROS: listNumbers -> lista de n meros aleatórios entre
#              0 e 1
#              listClientes -> lista de clientes
# RETURN: lista com os clientes que são atendidos no Posto C e
#          passam pela 3. fase e regressam 2. fase
def getClientesPostoC3FaseRegressam(listNumbers, listClientes):
    listaClientes = []
    bkList = list(listClientes)

    for num in listNumbers:
        # 40% regressam 2 fase
        if num >= 0 and num <= 0.4:
            cliente = random.choice(list(bkList))
            listaClientes.append(cliente)
            bkList.remove(cliente)

    return listaClientes

##### RESULTADOS #####

# lista com os nClientes
listaClientes = []
for i in range(0, nClientes):
    listaClientes.append(i)

# lista com n meros aleatórios entre 0 e 1
listaNumerosEntre0e1 = getListNumerosEntre0e1(12, nClientes)

# lista com os tempos de chegada
temposChegada = getTemposChegada(listaNumerosEntre0e1)
temposChegada.sort()
#print(temposChegada)

# lista com os clientes prioritários (aleatórios entre 0 e
# nClientes-1)
listaNumerosEntre0e1 = getListNumerosEntre0e1(13, nClientes)
clientesPrioritarios = getClientesPrioritarios(
    listaNumerosEntre0e1, listaClientes)
clientesPrioritarios.sort()

```

```
# lista com os clientes gerais
clientesGerais = list(set(listaClientes) - set(
    clientesPrioritarios))
clientesGerais.sort()

# lista com os tempos de triagem
listaNumerosEntre0e1 = getListNumerosEntre0e1(10,nClientes)
temposTriagem = getTemposTriagem(listaNumerosEntre0e1)
temposTriagemBackup = temposTriagem
#print(temposTriagem)

# lista com os clientes que passam da 1. para a 3. fase
listaNumerosEntre0e1 = getListNumerosEntre0e1(10,nClientes)
clientes1para3fase = getCientes1para3Fase(listaNumerosEntre0e1 ,
    listaClientes)
clientes1para3fase.sort()

# lista com os clientes que passam na segunda fase
clientes2fase = list(set(listaClientes) - set(clientes1para3fase
))
clientes2fase.sort()

# lista com os clientes que s o atendidos no Posto A
listaNumerosEntre0e1 = getListNumerosEntre0e1(13,len(
    clientes2fase))
clientesPostoA = getCientesPostoA(listaNumerosEntre0e1 ,
    clientes2fase)
clientesPostoA.sort()

# lista com os clientes que s o atendidos no Posto C
clientes2faseMenosPostoA = list(set(clientes2fase) - set(
    clientesPostoA))
listaNumerosEntre0e1 = getListNumerosEntre0e1(12,len(
    clientes2faseMenosPostoA))
clientesPostoC = getCientesPostoC(listaNumerosEntre0e1 ,
    clientes2faseMenosPostoA)
clientesPostoC.sort()

# lista com os clientes que s o atendidos no Posto B
clientesPostoB = list(set(clientes2fase) - set(clientesPostoC+
    clientesPostoA))
clientesPostoB.sort()
```

```
# lista dos clientes que s o atendidos no PostoA, passam para a
3. fase
listaNumerosEntre0e1 = getListNumerosEntre0e1(13,len(
    clientesPostoA))
clientesPostoA3Fase = getClientesPostoA3Fase(
    listaNumerosEntre0e1 , clientesPostoA)

# lista dos clientes que s o atendidos no PostoA e que
abandonam o sistema
clientesPostoAAbandonamSistema = list(set(clientesPostoA) - set(
    clientesPostoA3Fase))

# lista dos clientes que s o atendidos no PostoA, passam para a
3. fase e regressam 2. fase
listaNumerosEntre0e1 = getListNumerosEntre0e1(12,len(
    clientesPostoA3Fase))
clientesPostoA3FaseRegressam = getClientesPostoA3FaseRegressam(
    listaNumerosEntre0e1 , clientesPostoA3Fase)

# lista dos clientes que s o atendidos no PostoA, passam para a
3. fase e abandonam o sistema
clientesPostoA3FaseAbandonam = list(set(clientesPostoA3Fase) -
    set(clientesPostoA3FaseRegressam))

# lista dos clientes que s o atendidos no PostoB, passam para a
3. fase
listaNumerosEntre0e1 = getListNumerosEntre0e1(11,len(
    clientesPostoB))
clientesPostoB3Fase = getClientesPostoB3Fase(
    listaNumerosEntre0e1 , clientesPostoB)

# lista dos clientes que s o atendidos no PostoB e que
abandonam o sistema
clientesPostoBAbandonamSistema = list(set(clientesPostoB) - set(
    clientesPostoB3Fase))

# lista dos clientes que s o atendidos no PostoB, passam para a
3. fase e regressam 2. fase
listaNumerosEntre0e1 = getListNumerosEntre0e1(12,len(
    clientesPostoB3Fase))
clientesPostoB3FaseRegressam = getClientesPostoB3FaseRegressam(
    listaNumerosEntre0e1 , clientesPostoB3Fase)
```

```
# lista dos clientes que s o atendidos no PostoB , passam para a
# 3. fase e abandonam o sistema
clientesPostoB3FaseAbandonam = list(set(clientesPostoB3Fase) -
set(clientesPostoB3FaseRegressam))

# lista dos clientes que s o atendidos no PostoC , passam para a
# 3. fase
listaNumerosEntre0e1 = getListNumerosEntre0e1(11,len(
clientesPostoC))
clientesPostoC3Fase = getClientesPostoC3Fase(
listaNumerosEntre0e1 , clientesPostoC)

# lista dos clientes que s o atendidos no PostoC e que
# abandonam o sistema
clientesPostoCAbandonamSistema = list(set(clientesPostoC) - set(
clientesPostoC3Fase))

# lista dos clientes que s o atendidos no PostoC , passam para a
# 3. fase e regressam 2. fase
listaNumerosEntre0e1 = getListNumerosEntre0e1(12,len(
clientesPostoC3Fase))
clientesPostoC3FaseRegressam = getClientesPostoC3FaseRegressam(
listaNumerosEntre0e1 , clientesPostoC3Fase)

# lista dos clientes que s o atendidos no PostoC , passam para a
# 3. fase e abandonam o sistema
clientesPostoC3FaseAbandonam = list(set(clientesPostoC3Fase) -
set(clientesPostoC3FaseRegressam))

def getTempos3fase(listNumbers):
    listTempos3fase = []

    for num in listNumbers:
        if num >= 0 and num <= 0.4:
            listTempos3fase.append(random.randrange(1,60))
        elif num >= 0.56 and num <= 0.95:
            listTempos3fase.append(random.randrange(60,120))
        else:
            listTempos3fase.append(random.randrange(120,180))

    return listTempos3fase
```

```
listaNumerosEntre0e1 = getListNumerosEntre0e1(10, len(
    clientes1para3fase)+len(clientesPostoA3Fase)+len(
    clientesPostoB3Fase)+len(clientesPostoC3Fase))
tempos3fase = getTempos3fase(listaNumerosEntre0e1)

def getTempos2fase(listNumbers, tempo1, tempo2, tempo3, tempo4, perc1
, perc2, perc3):
    listTempos2fase = []

    for num in listNumbers:
        if num >= 0 and num <= perc1:
            listTempos2fase.append(random.randrange(1, tempo1))
        elif num > perc1 and num <= perc2:
            listTempos2fase.append(random.randrange(tempo1,
            tempo2))
        elif num > perc2 and num <= perc3:
            listTempos2fase.append(random.randrange(tempo2,
            tempo3))
        else:
            listTempos2fase.append(random.randrange(tempo3,
            tempo4))

    return listTempos2fase

listaNumerosEntre0e1 = getListNumerosEntre0e1(10, len(
    clientesPostoA))
tempos2faseA = getTempos2fase(listaNumerosEntre0e1
, 300, 900, 1500, 1800, 0.25, 0.6, 0.9)

listaNumerosEntre0e1 = getListNumerosEntre0e1(10, len(
    clientesPostoB))
tempos2faseB = getTempos2fase(listaNumerosEntre0e1
, 300, 600, 900, 1200, 0.25, 0.7, 0.95)

listaNumerosEntre0e1 = getListNumerosEntre0e1(10, len(
    clientesPostoC))
tempos2faseC = getTempos2fase(listaNumerosEntre0e1
, 300, 600, 900, 1200, 0.1, 0.45, 0.9)
```

```
#Classe Cliente
class Cliente:
    def __init__(cliente, id, chegada, tatendimento, tipoassunto,
                 tatend2fase1, tatend2fase2, tatend3fase):
        cliente.id = id
        cliente.chegada = chegada
        cliente.tatendimento = tatendimento
        cliente.tipoassunto = tipoassunto
        cliente.tatend2fase1 = tatend2fase1
        cliente.tatend2fase2 = tatend2fase2
        cliente.tatend3fase = tatend3fase

listaUtentes = []
tabelaClientes = []

# Gera o dos Clientes do Sistema
for i in range(0,nClientes):
    dados = []
    if i in clientesGerais:
        dados.append("G"+str(i+1))
    else:
        dados.append("P"+str(i+1))
    dados.append(temposChegada[i])
    randomTempoTriagem = random.choice(list(temposTriagem))
    dados.append(randomTempoTriagem)
    temposTriagem.remove(randomTempoTriagem)
    if i in clientes2fase:
        if i in clientesPostoA:
            dados.append("A")
            randomTempo2fase = random.choice(list(tempos2faseA))
            dados.append(randomTempo2fase)
            tempos2faseA.remove(randomTempo2fase)
            if (i in clientesPostoA3FaseRegressam):
                num = random.randrange(60,1800)
                dados.append(num)
            else:
                dados.append(0)
            if (i in clientesPostoA3Fase):
                randomTempo3fase = random.choice(list(
                    tempos3fase))
                dados.append(randomTempo3fase)
                tempos3fase.remove(randomTempo3fase)
            else:
                dados.append(0)
        elif (i in clientesPostoB):
            dados.append("B")
```



```

        randomTempo2fase = random.choice(list(tempos2faseB))
        dados.append(randomTempo2fase)
        tempos2faseB.remove(randomTempo2fase)
        if i in clientesPostoB3FaseRegressam:
            num = random.randrange(60,1800)
            dados.append(num)
        else:
            dados.append(0)
        if (i in clientesPostoB3Fase):
            randomTempo3fase = random.choice(list(
                tempos3fase))
            dados.append(randomTempo3fase)
            tempos3fase.remove(randomTempo3fase)
        else:
            dados.append(0)
    elif (i in clientesPostoC):
        dados.append("C")
        randomTempo2fase = random.choice(list(tempos2faseC))
        dados.append(randomTempo2fase)
        tempos2faseC.remove(randomTempo2fase)
        if i in clientesPostoC3FaseRegressam:
            num = random.randrange(60,1800)
            dados.append(num)
        else:
            dados.append(0)
        if (i in clientesPostoC3Fase):
            randomTempo3fase = random.choice(list(
                tempos3fase))
            dados.append(randomTempo3fase)
            tempos3fase.remove(randomTempo3fase)
        else:
            dados.append(0)
    else:
        dados.append("-")
        dados.append(0)
        dados.append(0)
        randomTempo3fase = random.choice(list(tempos3fase))
        dados.append(randomTempo3fase)
        tempos3fase.remove(randomTempo3fase)
    tabelaClientes.append(dados)

for i in range(0,nClientes):
    listaUtentes.append(Cliente(tabelaClientes[i][0],
        tabelaClientes[i][1],tabelaClientes[i][2],tabelaClientes[
            i][3],tabelaClientes[i][4],tabelaClientes[i][5],
            tabelaClientes[i][6]))

```

6.2 Programa Principal e Rotinas de Simulação

```
from geracao_tempos_utentes import listaUtentes
import datetime

#//////////////////////////FUNCOES E ROTINAS DE
#SIMULACAO ////////////////////////////#
#OBJETIVO: inserir utente fila triagem
def inserirUtenteFilaTriagem(NUtente, clock, tatendimento):
    item = []
    item.append(NUtente)
    item.append(clock)
    item.append(NUtente[0])
    item.append(tatendimento)
    filaTriagem.append(item)

#OBJETIVO: inserir utente fila PostoA
def inserirUtenteFilaPostoA(idUtente, clock, tatendimento):
    item = []
    item.append(idUtente)
    item.append(clock)
    item.append(idUtente[0])
    item.append(tatendimento)
    filaPostoA.append(item)

#OBJETIVO: inserir utente fila PostoA2Vez
def inserirUtenteFilaPostoA2Vez(idUtente, clock, tatendimento):
    item = []
    item.append(idUtente)
    item.append(clock)
    item.append(idUtente[0])
    item.append(tatendimento)
    filaPostoA2Vez.append(item)

#OBJETIVO: inserir utente fila PostoB
def inserirUtenteFilaPostoB(idUtente, clock, tatendimento):
    item = []
    item.append(idUtente)
    item.append(clock)
    item.append(idUtente[0])
    item.append(tatendimento)
    filaPostoB.append(item)

#OBJETIVO: inserir utente fila PostoB2Vez
def inserirUtenteFilaPostoB2Vez(idUtente, clock, tatendimento):
    item = []
    item.append(idUtente)
    item.append(clock)
```

```
    item.append(idUtente[0])
    item.append(tatendimento)
    filaPostoB2Vez.append(item)

#OBJETIVO: inserir utente fila PostoC
def inserirUtenteFilaPostoC(idUtente, clock, tatendimento):
    item = []
    item.append(idUtente)
    item.append(clock)
    item.append(idUtente[0])
    item.append(tatendimento)
    filaPostoC.append(item)

#OBJETIVO: inserir utente fila PostoC2Vez
def inserirUtenteFilaPostoC2Vez(idUtente, clock, tatendimento):
    item = []
    item.append(idUtente)
    item.append(clock)
    item.append(idUtente[0])
    item.append(tatendimento)
    filaPostoC2Vez.append(item)

#OBJETIVO: inserir utente fila Tesouraria
def inserirUtenteFilaTesouraria(idUtente, clock, tatendimento):
    item = []
    item.append(idUtente)
    item.append(clock)
    item.append(idUtente[0])
    item.append(tatendimento)
    filaTesouraria.append(item)

#OBJETIVO: remover utentes da fila de espera da triagem
def removerUtenteFilaTriagem(indice):
    del filaTriagem[indice]

#OBJETIVO: remover utentes da fila de espera do PostoA
def removerUtenteFilaPostoA(indice):
    del filaPostoA[indice]

#OBJETIVO: remover utentes da fila de espera do PostoA2Vez
def removerUtenteFilaPostoA2Vez(indice):
    del filaPostoA2Vez[indice]

#OBJETIVO: remover utentes da fila de espera do PostoB
def removerUtenteFilaPostoB(indice):
    del filaPostoB[indice]

#OBJETIVO: remover utentes da fila de espera do PostoB2Vez
def removerUtenteFilaPostoB2Vez(indice):
```

```

    del filaPostoB2Vez[indice]

#OBJETIVO: remover utentes da fila de espera do PostoC
def removerUtenteFilaPostoC(indice):
    del filaPostoC[indice]

#OBJETIVO: remover utentes da fila de espera 2 do PostoC
def removerUtenteFilaPostoC2Vez(indice):
    del filaPostoC2Vez[indice]

#OBJETIVO: remover utentes da fila de espera da Tesouraria
def removerUtenteFilaTesouraria(indice):
    del filaTesouraria[indice]

#OBJETIVO: obter o tempo do clock e o tipo de evento da pr xima
            itera o
def gestaoTempo(TPChegada, TPTriagem, TPPostoA1, TPPostoA2,
TPPostoB1, TPPostoB2, TPPostoC, TPTesouraria):
    clock = min(TPChegada, TPTriagem, TPPostoA1, TPPostoA2,
        TPPostoB1, TPPostoB2, TPPostoC, TPTesouraria)

    if (clock == TPChegada):
        tipoEvento = 0 #chegada
    elif (clock == TPTriagem):
        tipoEvento = 1 #partida triagem
    elif (clock == TPPostoA1):
        tipoEvento = 2 #partida PostoA1
    elif (clock == TPPostoA2):
        tipoEvento = 3 #partida PostoA2
    elif (clock == TPPostoB1):
        tipoEvento = 4 #partida PostoB1
    elif (clock == TPPostoB2):
        tipoEvento = 5 #partida PostoB2
    elif (clock == TPPostoC):
        tipoEvento = 6 #partida PostoC
    elif (clock == TPTesouraria):
        tipoEvento = 7 #partida Tesouraria

    if (TPChegada == INFINITO and TPTriagem == INFINITO and
        TPPostoA1 == INFINITO and TPPostoA2 == INFINITO and
        TPPostoB1 == INFINITO and TPPostoB2 == INFINITO and
        TPPostoC == INFINITO and TPTesouraria == INFINITO):
        tipoEvento = -1 #fim da simula o

    return [clock, tipoEvento]

#OBJETIVO: evento de chegada de um utente ao sistema
def eventoChegada(clock, listaUtentes, n):
    #Variables

```

```

global TPChegada
global ETriagem
global TUFTriagem
global NUSistema
global TATriagem
global TPTriagem
global TTOcupacaoTriagem
global TTOcupacaoTriagem0911
global TTOcupacaoTriagem1113
global TTOcupacaoTriagem1315
global TTOcupacaoTriagem1517
global UtentesTriagem

if (n+1 == len(listaUtentes)):
    TPChegada = INFINITO
else:
    TPChegada = listaUtentes[n+1].tchegada
NUtente = listaUtentes[n].id
if (ETriagem == 1): #ocupado
    inserirUtenteFilaTriagem(NUtente, clock, listaUtentes[n]
        ].tatendimento)
    TUFTriagem = TUFTriagem + 1
else:
    NUSistema = NUSistema + 1
    ETriagem = 1 #ocupado
    UtentesTriagem.append(NUtente)
    TATriagem = listaUtentes[n].tatendimento
    TPTriagem = clock + TATriagem
    TTOcupacaoTriagem = TTOcupacaoTriagem + TATriagem
    if (clock >= 0 and clock <= 7200):
        TTOcupacaoTriagem0911 = TTOcupacaoTriagem0911 +
            TATriagem
    elif (clock > 7200 and clock <= 14400):
        TTOcupacaoTriagem1113 = TTOcupacaoTriagem1113 +
            TATriagem
    elif (clock > 14400 and clock <= 21600):
        TTOcupacaoTriagem1315 = TTOcupacaoTriagem1315 +
            TATriagem
    else:
        TTOcupacaoTriagem1517 = TTOcupacaoTriagem1517 +
            TATriagem

#OBJETIVO: evento de partida do posto de triagem
def eventoPartidaTriagem(clock, listaUtentes):
    #Variables
    global TPChegada
    global ETriagem
    global TUFTriagem
    global NUSistema

```

```
global TATriagem
global TPTriagem
global TTOcupacaoTriagem
global TTOcupacaoTriagem0911
global TTOcupacaoTriagem1113
global TTOcupacaoTriagem1315
global TTOcupacaoTriagem1517
global TTEsperaTriagem
global TTEsperaTriagem0911
global TTEsperaTriagem1113
global TTEsperaTriagem1315
global TTEsperaTriagem1517
global UtentesTriagem

prioritarios = []
gerais = []
prioritariosExist = False

utente2Fase = UtentesTriagem[0]
del UtentesTriagem[0]

if (filaTriagem == []):
    ETriagem = 0 #livre
    TPTriagem = INFINITO
else:
    for item in filaTriagem:
        if item.__contains__("P"):
            prioridades.append(filaTriagem.index(item))
            prioridadesExist = True
        else:
            gerais.append(filaTriagem.index(item))
    if (prioritariosExist):
        indice = min(prioritarios)
    else:
        indice = min(gerais)
    TChegada = filaTriagem[indice][1]
    TEsperaUtente = clock - TChegada
    NUSistema = NUSistema + 1
    UtentesTriagem.append(filaTriagem[indice][0])
    TATriagem = filaTriagem[indice][3]
    TPTriagem = clock + TATriagem
    removerUtenteFilaTriagem(indice)
    TTEsperaTriagem = TTEsperaTriagem + TEsperaUtente
    TTOcupacaoTriagem = TTOcupacaoTriagem + TATriagem
    if (clock >= 0 and clock <= 7200):
        TTOcupacaoTriagem0911 = TTOcupacaoTriagem0911 +
            TATriagem
        TTEsperaTriagem0911 = TTEsperaTriagem0911 +
            TEsperaUtente
```

```

    elif (clock > 7200 and clock <= 14400):
        TTOcupacaoTriagem1113 = TTOcupacaoTriagem1113 +
            TATriagem
        TTEsperaTriagem1113 = TTEsperaTriagem1113 +
            TEsperaUtente
    elif (clock > 14400 and clock <= 21600):
        TTOcupacaoTriagem1315 = TTOcupacaoTriagem1315 +
            TATriagem
        TTEsperaTriagem1315 = TTEsperaTriagem1315 +
            TEsperaUtente
    else:
        TTOcupacaoTriagem1517 = TTOcupacaoTriagem1517 +
            TATriagem
        TTEsperaTriagem1517 = TTEsperaTriagem1517 +
            TEsperaUtente

for item in listaUtentes:
    if item.id.__contains__(utente2Fase):
        tipoAssunto = item.tipoassunto
        indice = listaUtentes.index(item)

    if (tipoAssunto == 'A'):
        eventoChegadaPostoA(indice, clock, 1)
    elif (tipoAssunto == 'B'):
        eventoChegadaPostoB(indice, clock, 1)
    elif (tipoAssunto == 'C'):
        eventoChegadaPostoC(indice, clock, 1)
    elif (tipoAssunto == '-'):
        eventoChegadaTesouraria(indice, clock)

#OBJETIVO: evento de chegada ao posto A1
def eventoChegadaPostoA(indexUtente, clock, nvezes):
    #vari veis
    global EPostoA1
    global EPostoA2
    global TUFPostoA
    global TAPostA2
    global TPPostA2
    global TTOcupacaoPostoA2
    global TTOcupacaoPostoA20911
    global TTOcupacaoPostoA21113
    global TTOcupacaoPostoA21315
    global TTOcupacaoPostoA21517
    global TAPostA1
    global TPPostA1
    global TTOcupacaoPostoA1
    global TTOcupacaoPostoA10911
    global TTOcupacaoPostoA11113
    global TTOcupacaoPostoA11315

```

```

global TTOcupacaoPostoA11517
global NUPostoA
global UtentesPostoA1
global UtentesPostoA2

if (EPostoA1 == 1): #ocupado
    if (EPostoA2 == 1): #ocupado
        if (nvezes == 2): #utente que regressa ao PostoA
            inserirUtenteFilaPostoA2Vez(listaUtentes[
                indexUtente].id, clock, listaUtentes[
                indexUtente].tatend2fase2)
            TUFPostoA = TUFPostoA + 1
        else:
            inserirUtenteFilaPostoA(listaUtentes[indexUtente]
                .id, clock, listaUtentes[indexUtente].
                tatend2fase1)
            TUFPostoA = TUFPostoA + 1
    else:
        if (nvezes == 2): #utente que regressa ao PostoA
            EPostoA2 = 1 #ocupado
            TAPostoA2 = listaUtentes[indexUtente].
                tatend2fase2
            TPPostoA2 = clock + TAPostoA2
            TTOcupacaoPostoA2 = TTOcupacaoPostoA2 +
                TAPostoA2
            if (clock >= 0 and clock <= 7200):
                TTOcupacaoPostoA20911 =
                    TTOcupacaoPostoA20911 + TAPostoA2
            elif (clock > 7200 and clock <= 14400):
                TTOcupacaoPostoA21113 =
                    TTOcupacaoPostoA21113 + TAPostoA2
            elif (clock > 14400 and clock <= 21600):
                TTOcupacaoPostoA21315 =
                    TTOcupacaoPostoA21315 + TAPostoA2
            else:
                TTOcupacaoPostoA21517 =
                    TTOcupacaoPostoA21517 + TAPostoA2
        else:
            EPostoA2 = 1 #ocupado
            NUPostoA = NUPostoA + 1
            UtentesPostoA2.append(listaUtentes[indexUtente].
                id)
            TAPostoA2 = listaUtentes[indexUtente].
                tatend2fase1
            TPPostoA2 = clock + TAPostoA2
            TTOcupacaoPostoA2 = TTOcupacaoPostoA2 +
                TAPostoA2
            if (clock >= 0 and clock <= 7200):
                TTOcupacaoPostoA20911 =

```



```

        TTOcupacaoPostoA20911 + TAPostoA2
    elif (clock > 7200 and clock <= 14400):
        TTOcupacaoPostoA21113 =
            TTOcupacaoPostoA21113 + TAPostoA2
    elif (clock > 14400 and clock <= 21600):
        TTOcupacaoPostoA21315 =
            TTOcupacaoPostoA21315 + TAPostoA2
    else:
        TTOcupacaoPostoA21517 =
            TTOcupacaoPostoA21517 + TAPostoA2
else:
    if (nvezes == 2): #utente que regressa ao PostoA
        EPostoA1 = 1 #ocupado
        TAPostoA1 = listaUtentes[indexUtente].tatend2fase2
        TPPostoA1 = clock + TAPostoA1
        TTOcupacaoPostoA1 = TTOcupacaoPostoA1 + TAPostoA1
        if (clock >= 0 and clock <= 7200):
            TTOcupacaoPostoA10911 = TTOcupacaoPostoA10911 +
                TAPostoA1
        elif (clock > 7200 and clock <= 14400):
            TTOcupacaoPostoA11113 = TTOcupacaoPostoA11113 +
                TAPostoA1
        elif (clock > 14400 and clock <= 21600):
            TTOcupacaoPostoA11315 = TTOcupacaoPostoA11315 +
                TAPostoA1
        else:
            TTOcupacaoPostoA11517 = TTOcupacaoPostoA11517 +
                TAPostoA1
    else:
        EPostoA1 = 1 #ocupado
        NUPostoA = NUPostoA + 1
        UtentesPostoA1.append(listaUtentes[indexUtente].id)
        TAPostoA1 = listaUtentes[indexUtente].tatend2fase1
        TPPostoA1 = clock + TAPostoA1
        TTOcupacaoPostoA1 = TTOcupacaoPostoA1 + TAPostoA1
        if (clock >= 0 and clock <= 7200):
            TTOcupacaoPostoA10911 = TTOcupacaoPostoA10911 +
                TAPostoA1
        elif (clock > 7200 and clock <= 14400):
            TTOcupacaoPostoA11113 = TTOcupacaoPostoA11113 +
                TAPostoA1
        elif (clock > 14400 and clock <= 21600):
            TTOcupacaoPostoA11315 = TTOcupacaoPostoA11315 +
                TAPostoA1
        else:
            TTOcupacaoPostoA11517 = TTOcupacaoPostoA11517 +
                TAPostoA1

```

#OBJETIVO: evento de partida do PostoA1

```
def eventoPartidaPostoA1 ( clock ) :  
    # variáveis  
    global EPostoA1  
    global TPPostoA1  
    global TAPostoA1  
    global TTEsperaPostoA  
    global TTEsperaPostoA0911  
    global TTEsperaPostoA1113  
    global TTEsperaPostoA1315  
    global TTEsperaPostoA1517  
    global TTOcupacaoPostoA1  
    global TTOcupacaoPostoA10911  
    global TTOcupacaoPostoA11113  
    global TTOcupacaoPostoA11315  
    global TTOcupacaoPostoA11517  
    global NUPostoA  
    global UtentesPostoA1  
  
    prioritarios = []  
    gerais = []  
    prioritariosExist = False  
    prioritarios2Vez = []  
    gerais2Vez = []  
    prioritariosExist2Vez = False  
    utente3Fase = ''  
  
    if ( len( UtentesPostoA1 ) != 0 ) :  
        utente3Fase = UtentesPostoA1 [ 0 ]  
        del UtentesPostoA1 [ 0 ]  
  
    if ( filaPostoA == [] ) :  
        if ( filaPostoA2Vez == [] ) :  
            EPostoA1 = 0 #livre  
            TPPostoA1 = INFINITO  
        else :  
            for item in filaPostoA2Vez :  
                if item.__contains__( "P" ) :  
                    prioritarios2Vez . append ( filaPostoA2Vez . index  
                                                ( item ) )  
                    prioritariosExist2Vez = True  
                else :  
                    gerais2Vez . append ( filaPostoA2Vez . index ( item )  
                                          )  
  
            if ( prioritariosExist2Vez ) :  
                indice = min ( prioritarios2Vez )  
            else :  
                indice = min ( gerais2Vez )
```

```

TChegada = filaPostoA2Vez[indice][1]
TEsperaUtente = clock - TChegada
TAPostoA1 = filaPostoA2Vez[indice][3]
TPPostoA1 = clock + TAPostoA1
removerUtenteFilaPostoA2Vez(indice)
TTEsperaPostoA = TTEsperaPostoA + TEsperaUtente
TTOcupacaoPostoA1 = TTOcupacaoPostoA1 + TAPostoA1
if (clock >= 0 and clock <= 7200):
    TTOcupacaoPostoA10911 = TTOcupacaoPostoA10911 +
        TAPostoA1
    TTEsperaPostoA0911 = TTEsperaPostoA0911 +
        TEsperaUtente
elif (clock > 7200 and clock <= 14400):
    TTOcupacaoPostoA11113 = TTOcupacaoPostoA11113 +
        TAPostoA1
    TTEsperaPostoA1113 = TTEsperaPostoA1113 +
        TEsperaUtente
elif (clock > 14400 and clock <= 21600):
    TTOcupacaoPostoA11315 = TTOcupacaoPostoA11315 +
        TAPostoA1
    TTEsperaPostoA1315 = TTEsperaPostoA1315 +
        TEsperaUtente
else:
    TTOcupacaoPostoA11517 = TTOcupacaoPostoA11517 +
        TAPostoA1
    TTEsperaPostoA1517 = TTEsperaPostoA1517 +
        TEsperaUtente
else:
    if (filaPostoA2Vez == []):
        for item in filaPostoA:
            if item.__contains__("P"):
                prioritarios.append(filaPostoA.index(item))
                prioritariosExist = True
            else:
                gerais.append(filaPostoA.index(item))

    if (prioritariosExist):
        indice = min(prioritarios)
    else:
        indice = min(gerais)

TChegada = filaPostoA[indice][1]
NUPostoA = NUPostoA + 1
UtentesPostoA1.append(filaPostoA[indice][0])
TEsperaUtente = clock - TChegada
TAPostoA1 = filaPostoA[indice][3]
TPPostoA1 = clock + TAPostoA1
removerUtenteFilaPostoA(indice)
TTEsperaPostoA = TTEsperaPostoA + TEsperaUtente

```

```

TTOcupacaoPostoA1 = TTOcupacaoPostoA1 + TAPostoA1
if (clock >= 0 and clock <= 7200):
    TTOcupacaoPostoA10911 = TTOcupacaoPostoA10911 +
        TAPostoA1
    TTEsperaPostoA0911 = TTEsperaPostoA0911 +
        TEsperaUtente
elif (clock > 7200 and clock <= 14400):
    TTOcupacaoPostoA11113 = TTOcupacaoPostoA11113 +
        TAPostoA1
    TTEsperaPostoA1113 = TTEsperaPostoA1113 +
        TEsperaUtente
elif (clock > 14400 and clock <= 21600):
    TTOcupacaoPostoA11315 = TTOcupacaoPostoA11315 +
        TAPostoA1
    TTEsperaPostoA1315 = TTEsperaPostoA1315 +
        TEsperaUtente
else:
    TTOcupacaoPostoA11517 = TTOcupacaoPostoA11517 +
        TAPostoA1
    TTEsperaPostoA1517 = TTEsperaPostoA1517 +
        TEsperaUtente
else:
    for item in filaPostoA:
        if item.__contains__("P"):
            prioritarios.append(filaPostoA.index(item))
            prioritariosExist = True

    if (prioritariosExist != True):
        for item in filaPostoA2Vez:
            if item.__contains__("P"):
                prioritarios2Vez.append(filaPostoA2Vez.
                    index(item))
                prioritariosExist2Vez = True
            else:
                gerais2Vez.append(filaPostoA2Vez.index(
                    item))

    if (prioritariosExist):
        indice = min(prioritarios)

        TChegada = filaPostoA[indice][1]
        NUPostoA = NUPostoA + 1
        UtentesPostoA1.append(filaPostoA[indice][0])
        TEsperaUtente = clock - TChegada
        TAPostoA1 = filaPostoA[indice][3]
        TPPostoA1 = clock + TAPostoA1
        removerUtenteFilaPostoA(indice)
        TTEsperaPostoA = TTEsperaPostoA + TEsperaUtente
        TTOcupacaoPostoA1 = TTOcupacaoPostoA1 +

```

```

        TAPostoA1
    if (clock >= 0 and clock <= 7200):
        TTOcupacaoPostoA10911 =
            TTOcupacaoPostoA10911 + TAPostoA1
        TTEsperaPostoA0911 = TTEsperaPostoA0911 +
            TEsperaUtente
    elif (clock > 7200 and clock <= 14400):
        TTOcupacaoPostoA11113 =
            TTOcupacaoPostoA11113 + TAPostoA1
        TTEsperaPostoA1113 = TTEsperaPostoA1113 +
            TEsperaUtente
    elif (clock > 14400 and clock <= 21600):
        TTOcupacaoPostoA11315 =
            TTOcupacaoPostoA11315 + TAPostoA1
        TTEsperaPostoA1315 = TTEsperaPostoA1315 +
            TEsperaUtente
    else:
        TTOcupacaoPostoA11517 =
            TTOcupacaoPostoA11517 + TAPostoA1
        TTEsperaPostoA1517 = TTEsperaPostoA1517 +
            TEsperaUtente
else:
    if (prioritariosExist2Vez):
        indice = min(prioritarios2Vez)
    else:
        indice = min(gerais2Vez)

    TChegada = filaPostoA2Vez[indice][1]
    TEsperaUtente = clock - TChegada
    TAPostoA1 = filaPostoA2Vez[indice][3]
    TPPostoA1 = clock + TAPostoA1
    removerUtenteFilaPostoA2Vez(indice)
    TTEsperaPostoA = TTEsperaPostoA + TEsperaUtente
    TTOcupacaoPostoA1 = TTOcupacaoPostoA1 +
        TAPostoA1
    if (clock >= 0 and clock <= 7200):
        TTOcupacaoPostoA10911 =
            TTOcupacaoPostoA10911 + TAPostoA1
        TTEsperaPostoA0911 = TTEsperaPostoA0911 +
            TEsperaUtente
    elif (clock > 7200 and clock <= 14400):
        TTOcupacaoPostoA11113 =
            TTOcupacaoPostoA11113 + TAPostoA1
        TTEsperaPostoA1113 = TTEsperaPostoA1113 +
            TEsperaUtente
    elif (clock > 14400 and clock <= 21600):
        TTOcupacaoPostoA11315 =
            TTOcupacaoPostoA11315 + TAPostoA1
        TTEsperaPostoA1315 = TTEsperaPostoA1315 +

```

```

        TEsperaUtente
    else:
        TTOcupacaoPostoA11517 =
            TTOcupacaoPostoA11517 + TAPostoA1
        TTEsperaPostoA1517 = TTEsperaPostoA1517 +
            TEsperaUtente

    if (utente3Fase != ''):
        for item in listaUtentes:
            if item.id.__contains__(utente3Fase):
                index = listaUtentes.index(item)

                if (listaUtentes[index].tatend3fase != 0):
                    eventoChegadaTesouraria(index, clock)

#OBJETIVO: evento de partida do PostoA2
def eventoPartidaPostoA2(clock):
    #vari veis
    global EPostoA2
    global TPPostoA2
    global TAPostoA2
    global TTEsperaPostoA
    global TTEsperaPostoA0911
    global TTEsperaPostoA1113
    global TTEsperaPostoA1315
    global TTEsperaPostoA1517
    global TTOcupacaoPostoA2
    global TTOcupacaoPostoA20911
    global TTOcupacaoPostoA21113
    global TTOcupacaoPostoA21315
    global TTOcupacaoPostoA21517
    global NUPostoA
    global UtentesPostoA2

    prioritarios = []
    gerais = []
    prioritariosExist = False
    prioritarios2Vez = []
    gerais2Vez = []
    prioritariosExist2Vez = False
    utente3Fase = ''

    if (len(UtentesPostoA2) != 0):
        utente3Fase = UtentesPostoA2[0]
        del UtentesPostoA2[0]

    if (filaPostoA == []):
        if (filaPostoA2Vez == []):
            EPostoA2 = 0 #livre

```

```

TPPostoA2 = INFINITO
else:
    for item in filaPostoA2Vez:
        if item.__contains__("P"):
            prioritarios2Vez.append(filaPostoA2Vez.index(
                item))
            prioritariosExist2Vez = True
        else:
            gerais2Vez.append(filaPostoA2Vez.index(item)
                )

    if (prioritariosExist2Vez):
        indice = min(prioritarios2Vez)
    else:
        indice = min(gerais2Vez)

    TChegada = filaPostoA2Vez[indice][1]
    TEsperaUtente = clock - TChegada
    TAPostoA2 = filaPostoA2Vez[indice][3]
    TPPostoA2 = clock + TAPostoA2
    removerUtenteFilaPostoA2Vez(indice)
    TTEsperaPostoA = TTEsperaPostoA + TEsperaUtente
    TTOcupacaoPostoA2 = TTOcupacaoPostoA2 + TAPostoA2
    if (clock >= 0 and clock <= 7200):
        TTOcupacaoPostoA20911 = TTOcupacaoPostoA20911 +
            TAPostoA2
        TTEsperaPostoA0911 = TTEsperaPostoA0911 +
            TEsperaUtente
    elif (clock > 7200 and clock <= 14400):
        TTOcupacaoPostoA21113 = TTOcupacaoPostoA21113 +
            TAPostoA2
        TTEsperaPostoA1113 = TTEsperaPostoA1113 +
            TEsperaUtente
    elif (clock > 14400 and clock <= 21600):
        TTOcupacaoPostoA21315 = TTOcupacaoPostoA21315 +
            TAPostoA2
        TTEsperaPostoA1315 = TTEsperaPostoA1315 +
            TEsperaUtente
    else:
        TTOcupacaoPostoA21517 = TTOcupacaoPostoA21517 +
            TAPostoA2
        TTEsperaPostoA1517 = TTEsperaPostoA1517 +
            TEsperaUtente
else:
    if (filaPostoA2Vez == []):
        for item in filaPostoA:
            if item.__contains__("P"):
                prioritarios.append(filaPostoA.index(item))
                prioritariosExist = True

```

```

        else:
            gerais.append(filaPostoA.index(item))

    if (prioritariosExist):
        indice = min(prioritarios)
    else:
        indice = min(gerais)

    TChegada = filaPostoA[indice][1]
    NUPostoA = NUPostoA + 1
    UtentesPostoA2.append(filaPostoA[indice][0])
    TEsperaUtente = clock - TChegada
    TAPostoA2 = filaPostoA[indice][3]
    TPPostoA2 = clock + TAPostoA2
    removerUtenteFilaPostoA(indice)
    TTEsperaPostoA = TTEsperaPostoA + TEsperaUtente
    TTOcupacaoPostoA2 = TTOcupacaoPostoA2 + TAPostoA2
    if (clock >= 0 and clock <= 7200):
        TTOcupacaoPostoA20911 = TTOcupacaoPostoA20911 +
            TAPostoA2
        TTEsperaPostoA0911 = TTEsperaPostoA0911 +
            TEsperaUtente
    elif (clock > 7200 and clock <= 14400):
        TTOcupacaoPostoA21113 = TTOcupacaoPostoA21113 +
            TAPostoA2
        TTEsperaPostoA1113 = TTEsperaPostoA1113 +
            TEsperaUtente
    elif (clock > 14400 and clock <= 21600):
        TTOcupacaoPostoA21315 = TTOcupacaoPostoA21315 +
            TAPostoA2
        TTEsperaPostoA1315 = TTEsperaPostoA1315 +
            TEsperaUtente
    else:
        TTOcupacaoPostoA21517 = TTOcupacaoPostoA21517 +
            TAPostoA2
        TTEsperaPostoA1517 = TTEsperaPostoA1517 +
            TEsperaUtente
    else:
        for item in filaPostoA:
            if item.__contains__("P"):
                prioridades.append(filaPostoA.index(item))
                prioridadesExist = True

    if (prioritariosExist != True):
        for item in filaPostoA2Vez:
            if item.__contains__("P"):
                prioridades2Vez.append(filaPostoA2Vez.
                    index(item))
                prioridadesExist2Vez = True

```



```

        else:
            gerais2Vez.append(filaPostoA2Vez.index(
                item))

    if (prioritariosExist):
        indice = min(prioritarios)

        TChegada = filaPostoA[indice][1]
        NUPostoA = NUPostoA + 1
        UtentesPostoA2.append(filaPostoA[indice][0])
        TEsperaUtente = clock - TChegada
        TAPostoA2 = filaPostoA[indice][3]
        TPPostoA2 = clock + TAPostoA2
        removerUtenteFilaPostoA(indice)
        TTEsperaPostoA = TTEsperaPostoA + TEsperaUtente
        TTOcupacaoPostoA2 = TTOcupacaoPostoA2 +
            TAPostoA2
        if (clock >= 0 and clock <= 7200):
            TTOcupacaoPostoA20911 =
                TTOcupacaoPostoA20911 + TAPostoA2
            TTEsperaPostoA0911 = TTEsperaPostoA0911 +
                TEsperaUtente
        elif (clock > 7200 and clock <= 14400):
            TTOcupacaoPostoA21113 =
                TTOcupacaoPostoA21113 + TAPostoA2
            TTEsperaPostoA1113 = TTEsperaPostoA1113 +
                TEsperaUtente
        elif (clock > 14400 and clock <= 21600):
            TTOcupacaoPostoA21315 =
                TTOcupacaoPostoA21315 + TAPostoA2
            TTEsperaPostoA1315 = TTEsperaPostoA1315 +
                TEsperaUtente
        else:
            TTOcupacaoPostoA21517 =
                TTOcupacaoPostoA21517 + TAPostoA2
            TTEsperaPostoA1517 = TTEsperaPostoA1517 +
                TEsperaUtente
    else:
        if (prioritariosExist2Vez):
            indice = min(prioritarios2Vez)
        else:
            indice = min(gerais2Vez)

        TChegada = filaPostoA2Vez[indice][1]
        TEsperaUtente = clock - TChegada
        TAPostoA2 = filaPostoA2Vez[indice][3]
        TPPostoA2 = clock + TAPostoA2
        removerUtenteFilaPostoA2Vez(indice)
        TTEsperaPostoA = TTEsperaPostoA + TEsperaUtente

```

```

        TTOcupacaoPostoA2 = TTOcupacaoPostoA2 +
            TAPostoA2
        if (clock >= 0 and clock <= 7200):
            TTOcupacaoPostoA20911 =
                TTOcupacaoPostoA20911 + TAPostoA2
            TTEsperaPostoA0911 = TTEsperaPostoA0911 +
                TEsperaUtente
        elif (clock > 7200 and clock <= 14400):
            TTOcupacaoPostoA21113 =
                TTOcupacaoPostoA21113 + TAPostoA2
            TTEsperaPostoA1113 = TTEsperaPostoA1113 +
                TEsperaUtente
        elif (clock > 14400 and clock <= 21600):
            TTOcupacaoPostoA21315 =
                TTOcupacaoPostoA21315 + TAPostoA2
            TTEsperaPostoA1315 = TTEsperaPostoA1315 +
                TEsperaUtente
        else:
            TTOcupacaoPostoA21517 =
                TTOcupacaoPostoA21517 + TAPostoA2
            TTEsperaPostoA1517 = TTEsperaPostoA1517 +
                TEsperaUtente

    if (utente3Fase != ''):
        for item in listaUtentes:
            if item.id.__contains__(utente3Fase):
                index = listaUtentes.index(item)

            if (listaUtentes[index].tatend3fase != 0):
                eventoChegadaTesouraria(index, clock)

#OBJETIVO: evento de chegada ao posto B
def eventoChegadaPostoB(indexUtente, clock, nvezes):
    #vari veis
    global EPostoB1
    global EPostoB2
    global TUFPostoB
    global TAPostoB2
    global TPPostoB2
    global TTOcupacaoPostoB2
    global TTOcupacaoPostoB20911
    global TTOcupacaoPostoB21113
    global TTOcupacaoPostoB21315
    global TTOcupacaoPostoB21517
    global TAPostoB1
    global TPPostoB1
    global TTOcupacaoPostoB1
    global TTOcupacaoPostoB10911
    global TTOcupacaoPostoB11113

```

```

global TTOcupacaoPostoB11315
global TTOcupacaoPostoB11517
global NUPostoB
global UtentesPostoB1
global UtentesPostoB2

if (EPostoB1 == 1): #ocupado
    if (EPostoB2 == 1): #ocupado
        if (nvezes == 2): #utente regressa ao PostoB
            inserirUtenteFilaPostoB2Vez(listaUtentes[
                indexUtente].id, clock, listaUtentes[
                indexUtente].tatend2fase2)
            TUFPostoB = TUFPostoB + 1
        else:
            inserirUtenteFilaPostoB(listaUtentes[indexUtente
                ].id, clock, listaUtentes[indexUtente].
                tatend2fase1)
            TUFPostoB = TUFPostoB + 1
    else:
        if (nvezes == 2): #utente regressa ao PostoB
            EPostoB2 = 1 #ocupado
            TAPostoB2 = listaUtentes[indexUtente].
                tatend2fase2
            TPPostoB2 = clock + TAPostoB2
            TTOcupacaoPostoB2 = TTOcupacaoPostoB2 +
                TAPostoB2
            if (clock >= 0 and clock <= 7200):
                TTOcupacaoPostoB20911 =
                    TTOcupacaoPostoB20911 + TAPostoB2
            elif (clock > 7200 and clock <= 14400):
                TTOcupacaoPostoB21113 =
                    TTOcupacaoPostoB21113 + TAPostoB2
            elif (clock > 14400 and clock <= 21600):
                TTOcupacaoPostoB21315 =
                    TTOcupacaoPostoB21315 + TAPostoB2
            else:
                TTOcupacaoPostoB21517 =
                    TTOcupacaoPostoB21517 + TAPostoB2
        else:
            EPostoB2 = 1 #ocupado
            NUPostoB = NUPostoB + 1
            UtentesPostoB2.append(listaUtentes[indexUtente].
                id)
            TAPostoB2 = listaUtentes[indexUtente].
                tatend2fase1
            TPPostoB2 = clock + TAPostoB2
            TTOcupacaoPostoB2 = TTOcupacaoPostoB2 +
                TAPostoB2
            if (clock >= 0 and clock <= 7200):

```

```

        TTOcupacaoPostoB20911 =
            TTOcupacaoPostoB20911 + TAPostoB2
    elif (clock > 7200 and clock <= 14400):
        TTOcupacaoPostoB21113 =
            TTOcupacaoPostoB21113 + TAPostoB2
    elif (clock > 14400 and clock <= 21600):
        TTOcupacaoPostoB21315 =
            TTOcupacaoPostoB21315 + TAPostoB2
    else:
        TTOcupacaoPostoB21517 =
            TTOcupacaoPostoB21517 + TAPostoB2
else:
    if (nvezes == 2): #utente regressa ao PostoB
        EPostoB1 = 1 #ocupado
        TAPostoB1 = listaUtentes[indexUtente].tatend2fase2
        TPPostoB1 = clock + TAPostoB1
        TTOcupacaoPostoB1 = TTOcupacaoPostoB1 + TAPostoB1
        if (clock >= 0 and clock <= 7200):
            TTOcupacaoPostoB10911 = TTOcupacaoPostoB10911 +
                TAPostoB1
        elif (clock > 7200 and clock <= 14400):
            TTOcupacaoPostoB11113 = TTOcupacaoPostoB11113 +
                TAPostoB1
        elif (clock > 14400 and clock <= 21600):
            TTOcupacaoPostoB11315 = TTOcupacaoPostoB11315 +
                TAPostoB1
        else:
            TTOcupacaoPostoB11517 = TTOcupacaoPostoB11517 +
                TAPostoB1
    else:
        EPostoB1 = 1 #ocupado
        NUPostoB = NUPostoB + 1
        UtentesPostoB1.append(listaUtentes[indexUtente].id)
        TAPostoB1 = listaUtentes[indexUtente].tatend2fase1
        TPPostoB1 = clock + TAPostoB1
        TTOcupacaoPostoB1 = TTOcupacaoPostoB1 + TAPostoB1
        if (clock >= 0 and clock <= 7200):
            TTOcupacaoPostoB10911 = TTOcupacaoPostoB10911 +
                TAPostoB1
        elif (clock > 7200 and clock <= 14400):
            TTOcupacaoPostoB11113 = TTOcupacaoPostoB11113 +
                TAPostoB1
        elif (clock > 14400 and clock <= 21600):
            TTOcupacaoPostoB11315 = TTOcupacaoPostoB11315 +
                TAPostoB1
        else:
            TTOcupacaoPostoB11517 = TTOcupacaoPostoB11517 +
                TAPostoB1

```

```

#OBJETIVO: evento de partida do PostoB1
def eventoPartidaPostoB1(clock):
    #vari veis
    global EPostoB1
    global TPPostoB1
    global TAPostoB1
    global TTEsperaPostoB
    global TTEsperaPostoB0911
    global TTEsperaPostoB1113
    global TTEsperaPostoB1315
    global TTEsperaPostoB1517
    global TTOcupacaoPostoB1
    global TTOcupacaoPostoB10911
    global TTOcupacaoPostoB11113
    global TTOcupacaoPostoB11315
    global TTOcupacaoPostoB11517
    global NUPostoB
    global UtentesPostoB1

    prioritarios = []
    gerais = []
    prioritariosExist = False
    prioritarios2Vez = []
    gerais2Vez = []
    prioritariosExist2Vez = False
    utente3Fase = ''

    if (len(UtentesPostoB1) != 0):
        utente3Fase = UtentesPostoB1[0]
        del UtentesPostoB1[0]

    if (filaPostoB == []):
        if (filaPostoB2Vez == []):
            EPostoB1 = 0 #livre
            TPPostoB1 = INFINITO
        else:
            for item in filaPostoB2Vez:
                if item.__contains__("P"):
                    prioritarios2Vez.append(filaPostoB2Vez.index(
                        item))
                    prioritariosExist2Vez = True
                else:
                    gerais2Vez.append(filaPostoB2Vez.index(item)
                    )

            if (prioritariosExist2Vez):
                indice = min(prioritarios2Vez)
            else:
                indice = min(gerais2Vez)

```

```

    TChegada = filaPostoB2Vez[indice][1]
    TEsperaUtente = clock - TChegada
    TAPostoB1 = filaPostoB2Vez[indice][3]
    TPPostoB1 = clock + TAPostoB1
    removerUtenteFilaPostoB2Vez(indice)
    TTEsperaPostoB = TTEsperaPostoB + TEsperaUtente
    TTOcupacaoPostoB1 = TTOcupacaoPostoB1 + TAPostoB1
    if (clock >= 0 and clock <= 7200):
        TTOcupacaoPostoB10911 = TTOcupacaoPostoB10911 +
            TAPostoB1
        TTEsperaPostoB0911 = TTEsperaPostoB0911 +
            TEsperaUtente
    elif (clock > 7200 and clock <= 14400):
        TTOcupacaoPostoB11113 = TTOcupacaoPostoB11113 +
            TAPostoB1
        TTEsperaPostoB1113 = TTEsperaPostoB1113 +
            TEsperaUtente
    elif (clock > 14400 and clock <= 21600):
        TTOcupacaoPostoB11315 = TTOcupacaoPostoB11315 +
            TAPostoB1
        TTEsperaPostoB1315 = TTEsperaPostoB1315 +
            TEsperaUtente
    else:
        TTOcupacaoPostoB11517 = TTOcupacaoPostoB11517 +
            TAPostoB1
        TTEsperaPostoB1517 = TTEsperaPostoB1517 +
            TEsperaUtente
else:
    if (filaPostoB2Vez == []):
        for item in filaPostoB:
            if item.__contains__("P"):
                prioritarios.append(filaPostoB.index(item))
                prioritariosExist = True
            else:
                gerais.append(filaPostoB.index(item))
        if (prioritariosExist):
            indice = min(prioritarios)
        else:
            indice = min(gerais)

    TChegada = filaPostoB[indice][1]
    NUPostoB = NUPostoB + 1
    UtentesPostoB1.append(filaPostoB[indice][0])
    TEsperaUtente = clock - TChegada
    TAPostoB1 = filaPostoB[indice][3]
    TPPostoB1 = clock + TAPostoB1
    removerUtenteFilaPostoB(indice)
    TTEsperaPostoB = TTEsperaPostoB + TEsperaUtente

```

```

TTOcupacaoPostoB1 = TTOcupacaoPostoB1 + TAPostoB1
if (clock >= 0 and clock <= 7200):
    TTOcupacaoPostoB10911 = TTOcupacaoPostoB10911 +
        TAPostoB1
    TTEsperaPostoB0911 = TTEsperaPostoB0911 +
        TEsperaUtente
elif (clock > 7200 and clock <= 14400):
    TTOcupacaoPostoB11113 = TTOcupacaoPostoB11113 +
        TAPostoB1
    TTEsperaPostoB1113 = TTEsperaPostoB1113 +
        TEsperaUtente
elif (clock > 14400 and clock <= 21600):
    TTOcupacaoPostoB11315 = TTOcupacaoPostoB11315 +
        TAPostoB1
    TTEsperaPostoB1315 = TTEsperaPostoB1315 +
        TEsperaUtente
else:
    TTOcupacaoPostoB11517 = TTOcupacaoPostoB11517 +
        TAPostoB1
    TTEsperaPostoB1517 = TTEsperaPostoB1517 +
        TEsperaUtente
else:
    for item in filaPostoB:
        if item.__contains__("P"):
            prioritarios.append(filaPostoB.index(item))
            prioritariosExist = True

    if (prioritariosExist != True):
        for item in filaPostoB2Vez:
            if item.__contains__("P"):
                prioritarios2Vez.append(filaPostoB2Vez.
                    index(item))
                prioritariosExist2Vez = True
            else:
                gerais2Vez.append(filaPostoB2Vez.index(
                    item))

    if (prioritariosExist):
        indice = min(prioritarios)

        TChegada = filaPostoB[indice][1]
        NUPostoB = NUPostoB + 1
        UtentesPostoB1.append(filaPostoB[indice][0])
        TEsperaUtente = clock - TChegada
        TAPostoB1 = filaPostoB[indice][3]
        TPPostoB1 = clock + TAPostoB1
        removerUtenteFilaPostoB(indice)
        TTEsperaPostoB = TTEsperaPostoB + TEsperaUtente
        TTOcupacaoPostoB1 = TTOcupacaoPostoB1 +

```

```

TAPostoB1
if (clock >= 0 and clock <= 7200):
    TTOcupacaoPostoB10911 =
        TTOcupacaoPostoB10911 + TAPostoB1
    TTEsperaPostoB0911 = TTEsperaPostoB0911 +
        TEsperaUtente
elif (clock > 7200 and clock <= 14400):
    TTOcupacaoPostoB11113 =
        TTOcupacaoPostoB11113 + TAPostoB1
    TTEsperaPostoB11113 = TTEsperaPostoB11113 +
        TEsperaUtente
elif (clock > 14400 and clock <= 21600):
    TTOcupacaoPostoB11315 =
        TTOcupacaoPostoB11315 + TAPostoB1
    TTEsperaPostoB1315 = TTEsperaPostoB1315 +
        TEsperaUtente
else:
    TTOcupacaoPostoB11517 =
        TTOcupacaoPostoB11517 + TAPostoB1
    TTEsperaPostoB1517 = TTEsperaPostoB1517 +
        TEsperaUtente
else:
    if (prioritariosExist2Vez):
        indice = min(prioritarios2Vez)
    else:
        indice = min(gerais2Vez)

TChegada = filaPostoB2Vez[indice][1]
TEsperaUtente = clock - TChegada
TAPostoB1 = filaPostoB2Vez[indice][3]
TPPostoB1 = clock + TAPostoB1
removerUtenteFilaPostoB2Vez(indice)
TTEsperaPostoB = TTEsperaPostoB + TEsperaUtente
TTOcupacaoPostoB1 = TTOcupacaoPostoB1 +
    TAPostoB1
if (clock >= 0 and clock <= 7200):
    TTOcupacaoPostoB10911 =
        TTOcupacaoPostoB10911 + TAPostoB1
    TTEsperaPostoB0911 = TTEsperaPostoB0911 +
        TEsperaUtente
elif (clock > 7200 and clock <= 14400):
    TTOcupacaoPostoB11113 =
        TTOcupacaoPostoB11113 + TAPostoB1
    TTEsperaPostoB11113 = TTEsperaPostoB11113 +
        TEsperaUtente
elif (clock > 14400 and clock <= 21600):
    TTOcupacaoPostoB11315 =
        TTOcupacaoPostoB11315 + TAPostoB1
    TTEsperaPostoB1315 = TTEsperaPostoB1315 +

```



```

        TEsperaUtente
    else:
        TTOcupacaoPostoB11517 =
            TTOcupacaoPostoB11517 + TAPostoB1
        TTEsperaPostoB1517 = TTEsperaPostoB1517 +
            TEsperaUtente

if (utente3Fase != ''):
    for item in listaUtentes:
        if item.id.__contains__(utente3Fase):
            index = listaUtentes.index(item)

            if (listaUtentes[index].tatend3fase != 0):
                eventoChegadaTesouraria(index, clock)

#OBJETIVO: evento de partida do PostoB2
def eventoPartidaPostoB2(clock):
    #vari veis
    global EPostoB2
    global TPPostoB2
    global TAPostoB2
    global TTEsperaPostoB
    global TTEsperaPostoB0911
    global TTEsperaPostoB1113
    global TTEsperaPostoB1315
    global TTEsperaPostoB1517
    global TTOcupacaoPostoB2
    global TTOcupacaoPostoB20911
    global TTOcupacaoPostoB21113
    global TTOcupacaoPostoB21315
    global TTOcupacaoPostoB21517
    global NUPostoB
    global UtentesPostoB2

    prioritarios = []
    gerais = []
    prioritariosExist = False
    prioritarios2Vez = []
    gerais2Vez = []
    prioritariosExist2Vez = False
    utente3Fase = ''

    if (len(UtentesPostoB2) != 0):
        utente3Fase = UtentesPostoB2[0]
        del UtentesPostoB2[0]

    if (filaPostoB == []):
        if (filaPostoB2Vez == []):
            EPostoB2 = 0 #livre

```

```

TPPostoB2 = INFINITO
else:
    for item in filaPostoB2Vez:
        if item.__contains__("P"):
            prioritarios2Vez.append(filaPostoB2Vez.index(
                item))
            prioritariosExist2Vez = True
        else:
            gerais2Vez.append(filaPostoB2Vez.index(item)
                )

    if (prioritariosExist2Vez):
        indice = min(prioritarios2Vez)
    else:
        indice = min(gerais2Vez)

    TChegada = filaPostoB2Vez[indice][1]
    TEsperaUtente = clock - TChegada
    TAPostoB2 = filaPostoB2Vez[indice][3]
    TPPostoB2 = clock + TAPostoB2
    removerUtenteFilaPostoB2Vez(indice)
    TTEsperaPostoB = TTEsperaPostoB + TEsperaUtente
    TTOcupacaoPostoB2 = TTOcupacaoPostoB2 + TAPostoB2
    if (clock >= 0 and clock <= 7200):
        TTOcupacaoPostoB20911 = TTOcupacaoPostoB20911 +
            TAPostoB2
        TTEsperaPostoB0911 = TTEsperaPostoB0911 +
            TEsperaUtente
    elif (clock > 7200 and clock <= 14400):
        TTOcupacaoPostoB21113 = TTOcupacaoPostoB21113 +
            TAPostoB2
        TTEsperaPostoB1113 = TTEsperaPostoB1113 +
            TEsperaUtente
    elif (clock > 14400 and clock <= 21600):
        TTOcupacaoPostoB21315 = TTOcupacaoPostoB21315 +
            TAPostoB2
        TTEsperaPostoB1315 = TTEsperaPostoB1315 +
            TEsperaUtente
    else:
        TTOcupacaoPostoB21517 = TTOcupacaoPostoB21517 +
            TAPostoB2
        TTEsperaPostoB1517 = TTEsperaPostoB1517 +
            TEsperaUtente
else:
    if (filaPostoB2Vez == []):
        for item in filaPostoB:
            if item.__contains__("P"):
                prioritarios.append(filaPostoB.index(item))
                prioritariosExist = True

```

```

        else:
            gerais.append(filaPostoB.index(item))
    if (prioritariosExist):
        indice = min(prioritarios)
    else:
        indice = min(gerais)
    TChegada = filaPostoB[indice][1]
    NUPostoB = NUPostoB + 1
    UtentesPostoB2.append(filaPostoB[indice][0])
    TEsperaUtente = clock - TChegada
    TAPostoB2 = filaPostoB[indice][3]
    TPPostoB2 = clock + TAPostoB2
    removerUtenteFilaPostoB(indice)
    TTEsperaPostoB = TTEsperaPostoB + TEsperaUtente
    TTOcupacaoPostoB2 = TTOcupacaoPostoB2 + TAPostoB2
    if (clock >= 0 and clock <= 7200):
        TTOcupacaoPostoB20911 = TTOcupacaoPostoB20911 +
            TAPostoB2
        TTEsperaPostoB0911 = TTEsperaPostoB0911 +
            TEsperaUtente
    elif (clock > 7200 and clock <= 14400):
        TTOcupacaoPostoB21113 = TTOcupacaoPostoB21113 +
            TAPostoB2
        TTEsperaPostoB1113 = TTEsperaPostoB1113 +
            TEsperaUtente
    elif (clock > 14400 and clock <= 21600):
        TTOcupacaoPostoB21315 = TTOcupacaoPostoB21315 +
            TAPostoB2
        TTEsperaPostoB1315 = TTEsperaPostoB1315 +
            TEsperaUtente
    else:
        TTOcupacaoPostoB21517 = TTOcupacaoPostoB21517 +
            TAPostoB2
        TTEsperaPostoB1517 = TTEsperaPostoB1517 +
            TEsperaUtente
else:
    for item in filaPostoB:
        if item.__contains__("P"):
            prioridades.append(filaPostoB.index(item))
            prioridadesExist = True

    if(prioritariosExist != True):
        for item in filaPostoB2Vez:
            if item.__contains__("P"):
                prioridades2Vez.append(filaPostoB2Vez.
                    index(item))
                prioridadesExist2Vez = True
            else:
                gerais2Vez.append(filaPostoB2Vez.index(

```

```

        item))

    if (prioritariosExist):
        indice = min(prioritarios)

        TChegada = filaPostoB[indice][1]
        NUPostoB = NUPostoB + 1
        UtentesPostoB2.append(filaPostoB[indice][0])
        TEsperaUtente = clock - TChegada
        TAPostoB2 = filaPostoB[indice][3]
        TPPostoB2 = clock + TAPostoB2
        removerUtenteFilaPostoB(indice)
        TTEsperaPostoB = TTEsperaPostoB + TEsperaUtente
        TTOcupacaoPostoB2 = TTOcupacaoPostoB2 +
            TAPostoB2
        if (clock >= 0 and clock <= 7200):
            TTOcupacaoPostoB20911 =
                TTOcupacaoPostoB20911 + TAPostoB2
            TTEsperaPostoB0911 = TTEsperaPostoB0911 +
                TEsperaUtente
        elif (clock > 7200 and clock <= 14400):
            TTOcupacaoPostoB21113 =
                TTOcupacaoPostoB21113 + TAPostoB2
            TTEsperaPostoB1113 = TTEsperaPostoB1113 +
                TEsperaUtente
        elif (clock > 14400 and clock <= 21600):
            TTOcupacaoPostoB21315 =
                TTOcupacaoPostoB21315 + TAPostoB2
            TTEsperaPostoB1315 = TTEsperaPostoB1315 +
                TEsperaUtente
        else:
            TTOcupacaoPostoB21517 =
                TTOcupacaoPostoB21517 + TAPostoB2
            TTEsperaPostoB1517 = TTEsperaPostoB1517 +
                TEsperaUtente
    else:
        if (prioritariosExist2Vez):
            indice = min(prioritarios2Vez)
        else:
            indice = min(gerais2Vez)

        TChegada = filaPostoB2Vez[indice][1]
        TEsperaUtente = clock - TChegada
        TAPostoB2 = filaPostoB2Vez[indice][3]
        TPPostoB2 = clock + TAPostoB2
        removerUtenteFilaPostoB2Vez(indice)
        TTEsperaPostoB = TTEsperaPostoB + TEsperaUtente
        TTOcupacaoPostoB2 = TTOcupacaoPostoB2 +
            TAPostoB2

```

```

        if (clock >= 0 and clock <= 7200):
            TTOcupacaoPostoB20911 =
                TTOcupacaoPostoB20911 + TAPostoB2
            TTEsperaPostoB0911 = TTEsperaPostoB0911 +
                TEsperaUtente
        elif (clock > 7200 and clock <= 14400):
            TTOcupacaoPostoB21113 =
                TTOcupacaoPostoB21113 + TAPostoB2
            TTEsperaPostoB1113 = TTEsperaPostoB1113 +
                TEsperaUtente
        elif (clock > 14400 and clock <= 21600):
            TTOcupacaoPostoB21315 =
                TTOcupacaoPostoB21315 + TAPostoB2
            TTEsperaPostoB1315 = TTEsperaPostoB1315 +
                TEsperaUtente
        else:
            TTOcupacaoPostoB21517 =
                TTOcupacaoPostoB21517 + TAPostoB2
            TTEsperaPostoB1517 = TTEsperaPostoB1517 +
                TEsperaUtente

    if (utente3Fase != ''):
        for item in listaUtentes:
            if item.id.__contains__(utente3Fase):
                index = listaUtentes.index(item)

                if (listaUtentes[index].tatend3fase != 0):
                    eventoChegadaTesouraria(index, clock)

#OBJETIVO: evento de chegada ao posto C
def eventoChegadaPostoC(indexUtente, clock, nvezes):
    #vari veis
    global EPostoC
    global TUFPostoC
    global TAPostoC
    global TPPostoC
    global TTOcupacaoPostoC
    global TTOcupacaoPostoC0911
    global TTOcupacaoPostoC1113
    global TTOcupacaoPostoC1315
    global TTOcupacaoPostoC1517
    global NUPostoC
    global UtentesPostoC

    if (EPostoC == 1): #ocupado
        if (nvezes == 2): #utente regressa ao PostoC
            inserirUtenteFilaPostoC2Vez(listaUtentes[indexUtente]
                .id, clock, listaUtentes[indexUtente].
                tatend2fase2)

```

```

        TUFPostoC = TUFPostoC + 1
    else:
        inserirUtenteFilaPostoC(listaUtentes[indexUtente].id
                                , clock, listaUtentes[indexUtente].tatend2fase1)
        TUFPostoC = TUFPostoC + 1
    else:
        if (nvezes == 2): #utente regressa ao PostoC
            EPostoC = 1 #ocupado
            TAPostoC = listaUtentes[indexUtente].tatend2fase2
            TPPostoC = clock + TAPostoC
            TTOcupacaoPostoC = TTOcupacaoPostoC + TAPostoC
            if (clock >= 0 and clock <= 7200):
                TTOcupacaoPostoC0911 = TTOcupacaoPostoC0911 +
                    TAPostoC
            elif (clock > 7200 and clock <= 14400):
                TTOcupacaoPostoC1113 = TTOcupacaoPostoC1113 +
                    TAPostoC
            elif (clock > 14400 and clock <= 21600):
                TTOcupacaoPostoC1315 = TTOcupacaoPostoC1315 +
                    TAPostoC
            else:
                TTOcupacaoPostoC1517 = TTOcupacaoPostoC1517 +
                    TAPostoC
        else:
            EPostoC = 1 #ocupado
            NUPostoC = NUPostoC + 1
            UtentesPostoC.append(listaUtentes[indexUtente].id)
            TAPostoC = listaUtentes[indexUtente].tatend2fase1
            TPPostoC = clock + TAPostoC
            TTOcupacaoPostoC = TTOcupacaoPostoC + TAPostoC
            if (clock >= 0 and clock <= 7200):
                TTOcupacaoPostoC0911 = TTOcupacaoPostoC0911 +
                    TAPostoC
            elif (clock > 7200 and clock <= 14400):
                TTOcupacaoPostoC1113 = TTOcupacaoPostoC1113 +
                    TAPostoC
            elif (clock > 14400 and clock <= 21600):
                TTOcupacaoPostoC1315 = TTOcupacaoPostoC1315 +
                    TAPostoC
            else:
                TTOcupacaoPostoC1517 = TTOcupacaoPostoC1517 +
                    TAPostoC

#OBJETIVO: evento de partida do PostoC
def eventoPartidaPostoC(clock):
    #vari veis
    global EPostoC
    global TPPostoC
    global TAPostoC

```

```

global TTEsperaPostoC
global TTEsperaPostoC0911
global TTEsperaPostoC1113
global TTEsperaPostoC1315
global TTEsperaPostoC1517
global TTOcupacaoPostoC
global TTOcupacaoPostoC0911
global TTOcupacaoPostoC1113
global TTOcupacaoPostoC1315
global TTOcupacaoPostoC1517
global NUPostoC
global UtentesPostoC

prioritarios = []
gerais = []
prioritarios2Vez = []
gerais2Vez = []
prioritariosExist = False
prioritariosExist2Vez = False
utente3Fase = ''

if (len(UtentesPostoC) != 0):
    utente3Fase = UtentesPostoC[0]
    del UtentesPostoC[0]

if (filaPostoC == []):
    if (filaPostoC2Vez == []):
        EPostoC = 0 #livre
        TPPostoC = INFINITO
    else:
        for item in filaPostoC2Vez:
            if (item.__contains__("P")):
                prioridades2Vez.append(filaPostoC2Vez.index(
                    item))
                prioridadesExist2Vez = True
            else:
                gerais2Vez.append(filaPostoC2Vez.index(item)
                    )

        if (prioritariosExist2Vez):
            indice = min(prioritarios2Vez)
        else:
            indice = min(gerais2Vez)

        TChegada = filaPostoC2Vez[indice][1]
        TEsperaUtente = clock - TChegada
        TAPostoC = filaPostoC2Vez[indice][3]
        TPPostoC = clock + TAPostoC
        removerUtenteFilaPostoC2Vez(indice)

```

```

TTEsperaPostoC = TTEsperaPostoC + TEsperaUtente
TTOcupacaoPostoC = TTOcupacaoPostoC + TAPostoC
if (clock >= 0 and clock <= 7200):
    TTOcupacaoPostoC0911 = TTOcupacaoPostoC0911 +
        TAPostoC
    TTEsperaPostoC0911 = TTEsperaPostoC0911 +
        TEsperaUtente
elif (clock > 7200 and clock <= 14400):
    TTOcupacaoPostoC1113 = TTOcupacaoPostoC1113 +
        TAPostoC
    TTEsperaPostoC1113 = TTEsperaPostoC1113 +
        TEsperaUtente
elif (clock > 14400 and clock <= 21600):
    TTOcupacaoPostoC1315 = TTOcupacaoPostoC1315 +
        TAPostoC
    TTEsperaPostoC1315 = TTEsperaPostoC1315 +
        TEsperaUtente
else:
    TTOcupacaoPostoC1517 = TTOcupacaoPostoC1517 +
        TAPostoC
    TTEsperaPostoC1517 = TTEsperaPostoC1517 +
        TEsperaUtente
else:
    if (filaPostoC2Vez == []):
        for item in filaPostoC:
            if item.__contains__("P"):
                prioritarios.append(filaPostoC.index(item))
                prioritariosExist = True
            else:
                gerais.append(filaPostoC.index(item))

    if (prioritariosExist):
        indice = min(prioritarios)
    else:
        indice = min(gerais)

    TChegada = filaPostoC[indice][1]
    NUPostoC = NUPostoC + 1
    UtentesPostoC.append(filaPostoC[indice][0])
    TEsperaUtente = clock - TChegada
    TAPostoC = filaPostoC[indice][3]
    TPPostoC = clock + TAPostoC
    removerUtenteFilaPostoC(indice)
    TTEsperaPostoC = TTEsperaPostoC + TEsperaUtente
    TTOcupacaoPostoC = TTOcupacaoPostoC + TAPostoC
    if (clock >= 0 and clock <= 7200):
        TTOcupacaoPostoC0911 = TTOcupacaoPostoC0911 +
            TAPostoC
        TTEsperaPostoC0911 = TTEsperaPostoC0911 +

```



```

        TEsperaUtente
    elif (clock > 7200 and clock <= 14400):
        TTOcupacaoPostoC1113 = TTOcupacaoPostoC1113 +
            TAPostoC
        TTEsperaPostoC1113 = TTEsperaPostoC1113 +
            TEsperaUtente
    elif (clock > 14400 and clock <= 21600):
        TTOcupacaoPostoC1315 = TTOcupacaoPostoC1315 +
            TAPostoC
        TTEsperaPostoC1315 = TTEsperaPostoC1315 +
            TEsperaUtente
    else:
        TTOcupacaoPostoC1517 = TTOcupacaoPostoC1517 +
            TAPostoC
        TTEsperaPostoC1517 = TTEsperaPostoC1517 +
            TEsperaUtente
else:
    for item in filaPostoC:
        if (item.__contains__("P")):
            prioritarios.append(filaPostoC.index(item))
            prioritariosExist = True

    if (prioritariosExist != True):
        for item in filaPostoC2Vez:
            if (item.__contains__("P")):
                prioritarios2Vez.append(filaPostoC2Vez.
                    index(item))
                prioritariosExist2Vez = True
            else:
                gerais2Vez.append(filaPostoC2Vez.index(
                    item))

    if (prioritariosExist):
        indice = min(prioritarios)

        TChegada = filaPostoC[indice][1]
        NUPostoC = NUPostoC + 1
        UtentesPostoC.append(filaPostoC[indice][0])
        TEsperaUtente = clock - TChegada
        TAPostoC = filaPostoC[indice][3]
        TPPostoC = clock + TAPostoC
        removerUtenteFilaPostoC(indice)
        TTEsperaPostoC = TTEsperaPostoC + TEsperaUtente
        TTOcupacaoPostoC = TTOcupacaoPostoC + TAPostoC
        if (clock >= 0 and clock <= 7200):
            TTOcupacaoPostoC0911 = TTOcupacaoPostoC0911
                + TAPostoC
            TTEsperaPostoC0911 = TTEsperaPostoC0911 +
                TEsperaUtente

```

```

elif (clock > 7200 and clock <= 14400):
    TTOcupacaoPostoC1113 = TTOcupacaoPostoC1113
    + TAPostoC
    TTEsperaPostoC1113 = TTEsperaPostoC1113 +
    TEsperaUtente
elif (clock > 14400 and clock <= 21600):
    TTOcupacaoPostoC1315 = TTOcupacaoPostoC1315
    + TAPostoC
    TTEsperaPostoC1315 = TTEsperaPostoC1315 +
    TEsperaUtente
else:
    TTOcupacaoPostoC1517 = TTOcupacaoPostoC1517
    + TAPostoC
    TTEsperaPostoC1517 = TTEsperaPostoC1517 +
    TEsperaUtente
else:
    if (prioritariosExist2Vez):
        indice = min(prioritarios2Vez)
    else:
        indice = min(gerais2Vez)

    TChegada = filaPostoC2Vez[indice][1]
    TEsperaUtente = clock - TChegada
    TAPostoC = filaPostoC2Vez[indice][3]
    TPPostoC = clock + TAPostoC
    removerUtenteFilaPostoC2Vez(indice)
    TTEsperaPostoC = TTEsperaPostoC + TEsperaUtente
    TTOcupacaoPostoC = TTOcupacaoPostoC + TAPostoC
    if (clock >= 0 and clock <= 7200):
        TTOcupacaoPostoC0911 = TTOcupacaoPostoC0911
        + TAPostoC
        TTEsperaPostoC0911 = TTEsperaPostoC0911 +
        TEsperaUtente
    elif (clock > 7200 and clock <= 14400):
        TTOcupacaoPostoC1113 = TTOcupacaoPostoC1113
        + TAPostoC
        TTEsperaPostoC1113 = TTEsperaPostoC1113 +
        TEsperaUtente
    elif (clock > 14400 and clock <= 21600):
        TTOcupacaoPostoC1315 = TTOcupacaoPostoC1315
        + TAPostoC
        TTEsperaPostoC1315 = TTEsperaPostoC1315 +
        TEsperaUtente
    else:
        TTOcupacaoPostoC1517 = TTOcupacaoPostoC1517
        + TAPostoC
        TTEsperaPostoC1517 = TTEsperaPostoC1517 +
        TEsperaUtente

```

```

if (utente3Fase != ''):
    for item in listaUtentes:
        if item.id.__contains__(utente3Fase):
            index = listaUtentes.index(item)

            if (listaUtentes[index].tatend3fase != 0):
                eventoChegadaTesouraria(index, clock)

#OBJETIVO: evento de chegada Tesouraria
def eventoChegadaTesouraria(indexUtente, clock):
    #vari veis
    global ETesouraria
    global TUFtesouraria
    global TATesouraria
    global TPTesouraria
    global TTOcupacaoTesouraria
    global TTOcupacaoTesouraria0911
    global TTOcupacaoTesouraria1113
    global TTOcupacaoTesouraria1315
    global TTOcupacaoTesouraria1517
    global NUTesouraria
    global UtentesTesouraria

    print(listaUtentes[indexUtente].id)

    if (ETesouraria == 1): #ocupado
        inserirUtenteFilaTesouraria(listaUtentes[indexUtente].id
            , clock, listaUtentes[indexUtente].tatend3fase)
        TUFtesouraria = TUFtesouraria + 1
    else:
        ETesouraria = 1 #ocupado
        NUTesouraria = NUTesouraria + 1
        UtentesTesouraria.append(listaUtentes[indexUtente].id)
        TATesouraria = listaUtentes[indexUtente].tatend3fase
        TPTesouraria = clock + TATesouraria
        TTOcupacaoTesouraria = TTOcupacaoTesouraria +
            TATesouraria
        if (clock >= 0 and clock <= 7200):
            TTOcupacaoTesouraria0911 = TTOcupacaoTesouraria0911
                + TATesouraria
        elif (clock > 7200 and clock <= 14400):
            TTOcupacaoTesouraria1113 = TTOcupacaoTesouraria1113
                + TATesouraria
        elif (clock > 14400 and clock <= 21600):
            TTOcupacaoTesouraria1315 = TTOcupacaoTesouraria1315
                + TATesouraria
        else:
            TTOcupacaoTesouraria1517 = TTOcupacaoTesouraria1517
                + TATesouraria

```

```

#OBJETIVO: evento de partida da Tesouraria
def eventoPartidaTesouraria(clock):
    #variáveis
    global ETesouraria
    global TPTesouraria
    global TATesouraria
    global TTEsperaTesouraria
    global TTEsperaTesouraria0911
    global TTEsperaTesouraria1113
    global TTEsperaTesouraria1315
    global TTEsperaTesouraria1517
    global TTOcupacaoTesouraria
    global TTOcupacaoTesouraria0911
    global TTOcupacaoTesouraria1113
    global TTOcupacaoTesouraria1315
    global TTOcupacaoTesouraria1517
    global NUTesouraria
    global UtentesTesouraria

    prioritarios = []
    gerais = []
    prioritariosExist = False

    utente2Fase2 = UtentesTesouraria[0]
    del UtentesTesouraria[0]

    if (filaTesouraria == []):
        ETesouraria = 0 #livre
        TPTesouraria = INFINITO
    else:
        for item in filaTesouraria:
            if item.__contains__("P"):
                prioritarios.append(filaTesouraria.index(item))
                prioritariosExist = True
            else:
                gerais.append(filaTesouraria.index(item))
        if (prioritariosExist):
            indice = min(prioritarios)
        else:
            indice = min(gerais)
        TChegada = filaTesouraria[indice][1]
        NUTesouraria = NUTesouraria + 1
        UtentesTesouraria.append(filaTesouraria[indice][0])
        TEsperaUtente = clock - TChegada
        TATesouraria = filaTesouraria[indice][3]
        TPTesouraria = clock + TATesouraria
        removerUtenteFilaTesouraria(indice)
        TTEsperaTesouraria = TTEsperaTesouraria + TEsperaUtente

```

```

TTOcupacaoTesouraria = TTOcupacaoTesouraria +
    TATesouraria
if (clock >= 0 and clock <= 7200):
    TTOcupacaoTesouraria0911 = TTOcupacaoTesouraria0911
        + TATesouraria
    TTEsperaTesouraria0911 = TTEsperaTesouraria0911 +
        TEsperaUtente
elif (clock > 7200 and clock <= 14400):
    TTOcupacaoTesouraria1113 = TTOcupacaoTesouraria1113
        + TATesouraria
    TTEsperaTesouraria1113 = TTEsperaTesouraria1113 +
        TEsperaUtente
elif (clock > 14400 and clock <= 21600):
    TTOcupacaoTesouraria1315 = TTOcupacaoTesouraria1315
        + TATesouraria
    TTEsperaTesouraria1315 = TTEsperaTesouraria1315 +
        TEsperaUtente
else:
    TTOcupacaoTesouraria1517 = TTOcupacaoTesouraria1517
        + TATesouraria
    TTEsperaTesouraria1517 = TTEsperaTesouraria1517 +
        TEsperaUtente

for item in listaUtentes:
    if item.id.__contains__(utente2Fase2):
        index = listaUtentes.index(item)
        tipoAssunto = listaUtentes[index].tipoassunto

if (listaUtentes[index].tatend2fase2 != 0):
    if (tipoAssunto == 'A'):
        eventoChegadaPostoA(index, clock, 2)
    elif (tipoAssunto == 'B'):
        eventoChegadaPostoB(index, clock, 2)
    elif (tipoAssunto == 'C'):
        eventoChegadaPostoC(index, clock, 2)

```

#

```

////////////////////////////////////

```

6.3 Programa Principal

```

#////////////////////////////////////// MAIN
#//////////////////////////////////////#
#Inicializa o de variaveis
INFINITO = 999999999
filaTriagem = []
filaPostoA = []
filaPostoB = []
filaPostoC = []
filaTesouraria = []
clock = 0
NUtente = ''
ETriagem = 0 #livre
TUFTriagem = 0
TATriagem = 0
TPChegada = listaUtentes[0].tchegada
TPTriagem = INFINITO
NUSistema = 0
TTEsperaTriagem = 0
TTEsperaTriagem0911 = 0
TTEsperaTriagem1113 = 0
TTEsperaTriagem1315 = 0
TTEsperaTriagem1517 = 0
TTOcupacaoTriagem = 0
TTOcupacaoTriagem0911 = 0
TTOcupacaoTriagem1113 = 0
TTOcupacaoTriagem1315 = 0
TTOcupacaoTriagem1517 = 0
nUtentes = 0
terminar = 0
UtentesTriagem = []
EPostoA1 = 0 #livre
EPostoA2 = 0 #livre
TUFPostoA = 0
TAPosttoA2 = 0
TPPosttoA2 = INFINITO
TTOcupacaoPosttoA2 = 0
TTOcupacaoPosttoA20911 = 0
TTOcupacaoPosttoA21113 = 0
TTOcupacaoPosttoA21315 = 0
TTOcupacaoPosttoA21517 = 0
TAPosttoA1 = 0
TPPosttoA1 = INFINITO
TTOcupacaoPosttoA1 = 0
TTOcupacaoPosttoA10911 = 0
TTOcupacaoPosttoA11113 = 0
TTOcupacaoPosttoA11315 = 0

```

```
TTocupacaoPostoA11517 = 0
TTEsperaPostoA = 0
TTEsperaPostoA0911 = 0
TTEsperaPostoA1113 = 0
TTEsperaPostoA1315 = 0
TTEsperaPostoA1517 = 0
NUPostoA = 0
EPostoB1 = 0 #livre
EPostoB2 = 0 #livre
TUFPostoB = 0
TAPostob2 = 0
TPPostoB2 = INFINITO
TTocupacaoPostoB2 = 0
TTocupacaoPostoB20911 = 0
TTocupacaoPostoB21113 = 0
TTocupacaoPostoB21315 = 0
TTocupacaoPostoB21517 = 0
TAPostob1 = 0
TPPostoB1 = INFINITO
TTocupacaoPostoB1 = 0
TTocupacaoPostoB10911 = 0
TTocupacaoPostoB11113 = 0
TTocupacaoPostoB11315 = 0
TTocupacaoPostoB11517 = 0
NUPostoB = 0
TTEsperaPostoB = 0
TTEsperaPostoB0911 = 0
TTEsperaPostoB1113 = 0
TTEsperaPostoB1315 = 0
TTEsperaPostoB1517 = 0
EPostoC = 0 #livre
TUFPostoC = 0
TAPostoc = 0
TPPostoC = INFINITO
TTocupacaoPostoC = 0
TTocupacaoPostoC0911 = 0
TTocupacaoPostoC1113 = 0
TTocupacaoPostoC1315 = 0
TTocupacaoPostoC1517 = 0
NUPostoC = 0
TTEsperaPostoC = 0
TTEsperaPostoC0911 = 0
TTEsperaPostoC1113 = 0
TTEsperaPostoC1315 = 0
TTEsperaPostoC1517 = 0
UtentesPostoC = []
UtentesPostoA1 = []
UtentesPostoA2 = []
UtentesPostoB1 = []
```

```

UtentesPostoB2 = []
TPTesouraria = INFINITO
ETesouraria = 0 #livre
TUFtesouraria = 0
TATesouraria = 0
TTOcupacaoTesouraria = 0
TTOcupacaoTesouraria0911 = 0
TTOcupacaoTesouraria1113 = 0
TTOcupacaoTesouraria1315 = 0
TTOcupacaoTesouraria1517 = 0
NUTesouraria = 0
UtentesTesouraria = []
TTEsperaTesouraria = 0
TTEsperaTesouraria0911 = 0
TTEsperaTesouraria1113 = 0
TTEsperaTesouraria1315 = 0
TTEsperaTesouraria1517 = 0
filaPostoC2Vez = []
filaPostoB2Vez = []
filaPostoA2Vez = []

while(True):
    getClockTipoEvento = gestaoTempo(TPChegada, TPTriagem,
        TPPostoA1, TPPostoA2, TPPostoB1, TPPostoB2, TPPostoC,
        TPTesouraria) #[0] - clock; [1] - tipoEvento

    if (getClockTipoEvento[1] == 0): #chegada
        nUtentes = nUtentes + 1
        eventoChegada(getClockTipoEvento[0], listaUtentes,
            nUtentes - 1)
        if (nUtentes == len(listaUtentes)):
            TPChegada = INFINITO
    elif (getClockTipoEvento[1] == 1): #partida triagem
        eventoPartidaTriagem(getClockTipoEvento[0], listaUtentes)
    elif (getClockTipoEvento[1] == 2): #partida PostoA1
        eventoPartidaPostoA1(getClockTipoEvento[0])
    elif (getClockTipoEvento[1] == 3): #partida PostoA2
        eventoPartidaPostoA2(getClockTipoEvento[0])
    elif (getClockTipoEvento[1] == 4): #partida PostoB1
        eventoPartidaPostoB1(getClockTipoEvento[0])
    elif (getClockTipoEvento[1] == 5): #partida PostoB2
        eventoPartidaPostoB2(getClockTipoEvento[0])
    elif (getClockTipoEvento[1] == 6): #partida PostoC
        eventoPartidaPostoC(getClockTipoEvento[0])
    elif (getClockTipoEvento[1] == 7): #partida Tesouraria
        eventoPartidaTesouraria(getClockTipoEvento[0])
    else: #fim da simula o
        terminar = 1

```



```

        if (terminar == 1):
            break
# //////////////////////////////////////// END MAIN
# //////////////////////////////////////// #

# Print statistics
print('///////////////////////////////// Servi o de Atendimento – FINAN AS
///////////////////////////////// ')
print('N.    Utentes Sistema: ', NUSistema)
print('Total Utentes Fila Triagem: ', TUFTriagem)
print('Tempo M dio Espera Fila Triagem (h:mm:ss): ', str(
    datetime.timedelta(seconds=round(TTEsperaTriagem/NUSistema,0)
)))
print('Tempo M dio Espera Fila Triagem (09h–11h): ', str(
    datetime.timedelta(seconds=round(TTEsperaTriagem0911/
    NUSistema,0))))
print('Tempo M dio Espera Fila Triagem (11h–13h): ', str(
    datetime.timedelta(seconds=round(TTEsperaTriagem1113/
    NUSistema,0))))
print('Tempo M dio Espera Fila Triagem (13h–15h): ', str(
    datetime.timedelta(seconds=round(TTEsperaTriagem1315/
    NUSistema,0))))
print('Tempo M dio Espera Fila Triagem (15h–17h): ', str(
    datetime.timedelta(seconds=round(TTEsperaTriagem1517/
    NUSistema,0))))
print('Tempo Total Ocupa o Posto Triagem: ', str(datetime.
    timedelta(seconds=TTOcupacaoTriagem)))
print('Tempo Total Ocupa o Posto Triagem (09h–11h): ', str(
    datetime.timedelta(seconds=TTOcupacaoTriagem0911)))
print('Tempo Total Ocupa o Posto Triagem (11h–13h): ', str(
    datetime.timedelta(seconds=TTOcupacaoTriagem1113)))
print('Tempo Total Ocupa o Posto Triagem (13h–15h): ', str(
    datetime.timedelta(seconds=TTOcupacaoTriagem1315)))
print('Tempo Total Ocupa o Posto Triagem (15h–17h): ', str(
    datetime.timedelta(seconds=TTOcupacaoTriagem1517)))
print('Total Utentes Posto A: ', NUPostoA)
print('Total Utentes Fila Posto A: ', TUFPostoA)
print('Tempo M dio Espera Fila Posto A: ', str(datetime.
    timedelta(seconds=round(TTEsperaPostoA/NUPostoA,0))))
print('Tempo M dio Espera Fila Posto A (09h–11h): ', str(
    datetime.timedelta(seconds=round(TTEsperaPostoA0911/NUPostoA
    ,0))))
print('Tempo M dio Espera Fila Posto A (11h–13h): ', str(
    datetime.timedelta(seconds=round(TTEsperaPostoA1113/NUPostoA
    ,0))))
print('Tempo M dio Espera Fila Posto A (13h–15h): ', str(
    datetime.timedelta(seconds=round(TTEsperaPostoA1315/NUPostoA
    ,0))))
print('Tempo M dio Espera Fila Posto A (15h–17h): ', str(

```

```

        datetime.timedelta(seconds=round(TTEsperaPostoA1517/NUPostoA
,0))))
print('Tempo Total Ocupa o Posto A1:', str(datetime.timedelta
(seconds=TTOcupacaoPostoA1)))
print('Tempo Total Ocupa o Posto A1 (09h-11h): ', str(datetime
.timedelta(seconds=TTOcupacaoPostoA10911)))
print('Tempo Total Ocupa o Posto A1 (11h-13h): ', str(datetime
.timedelta(seconds=TTOcupacaoPostoA11113)))
print('Tempo Total Ocupa o Posto A1 (13h-15h): ', str(datetime
.timedelta(seconds=TTOcupacaoPostoA11315)))
print('Tempo Total Ocupa o Posto A1 (15h-17h): ', str(datetime
.timedelta(seconds=TTOcupacaoPostoA11517)))
print('Tempo Total Ocupa o Posto A2:', str(datetime.timedelta
(seconds=TTOcupacaoPostoA2)))
print('Tempo Total Ocupa o Posto A2 (09h-11h): ', str(datetime
.timedelta(seconds=TTOcupacaoPostoA20911)))
print('Tempo Total Ocupa o Posto A2 (11h-13h): ', str(datetime
.timedelta(seconds=TTOcupacaoPostoA21113)))
print('Tempo Total Ocupa o Posto A2 (13h-15h): ', str(datetime
.timedelta(seconds=TTOcupacaoPostoA21315)))
print('Tempo Total Ocupa o Posto A2 (15h-17h): ', str(datetime
.timedelta(seconds=TTOcupacaoPostoA21517)))
print('Total Utentes Posto B:', NUPostoB)
print('Total Utentes Fila Posto B:', TUFPostoB)
print('Tempo M dio Espera Fila Posto B:', str(datetime.
timedelta(seconds=round(TTEsperaPostoB/NUPostoB,0))))
print('Tempo M dio Espera Fila Posto B (09h-11h): ', str(
datetime.timedelta(seconds=round(TTEsperaPostoB0911/NUPostoB
,0))))
print('Tempo M dio Espera Fila Posto B (11h-13h): ', str(
datetime.timedelta(seconds=round(TTEsperaPostoB1113/NUPostoB
,0))))
print('Tempo M dio Espera Fila Posto B (13h-15h): ', str(
datetime.timedelta(seconds=round(TTEsperaPostoB1315/NUPostoB
,0))))
print('Tempo M dio Espera Fila Posto B (15h-17h): ', str(
datetime.timedelta(seconds=round(TTEsperaPostoB1517/NUPostoB
,0))))
print('Tempo Total Ocupa o Posto B1:', str(datetime.timedelta
(seconds=TTOcupacaoPostoB1)))
print('Tempo Total Ocupa o Posto B1 (09h-11h): ', str(datetime
.timedelta(seconds=TTOcupacaoPostoB10911)))
print('Tempo Total Ocupa o Posto B1 (11h-13h): ', str(datetime
.timedelta(seconds=TTOcupacaoPostoB11113)))
print('Tempo Total Ocupa o Posto B1 (13h-15h): ', str(datetime
.timedelta(seconds=TTOcupacaoPostoB11315)))
print('Tempo Total Ocupa o Posto B1 (15h-17h): ', str(datetime
.timedelta(seconds=TTOcupacaoPostoB11517)))
print('Tempo Total Ocupa o Posto B2:', str(datetime.timedelta

```

```

        (seconds=TTOcupacaoPostoB2)))
print('Tempo Total Ocupa o Posto B2 (09h-11h): ', str(datetime
    .timedelta(seconds=TTOcupacaoPostoB20911)))
print('Tempo Total Ocupa o Posto B2 (11h-13h): ', str(datetime
    .timedelta(seconds=TTOcupacaoPostoB21113)))
print('Tempo Total Ocupa o Posto B2 (13h-15h): ', str(datetime
    .timedelta(seconds=TTOcupacaoPostoB21315)))
print('Tempo Total Ocupa o Posto B2 (15h-17h): ', str(datetime
    .timedelta(seconds=TTOcupacaoPostoB21517)))
print('Total Utentes Posto C:', NUPostoC)
print('Total Utentes Fila Posto C:', TUFPostoC)
print('Tempo M dio Espera Fila Posto C:', str(datetime.
    timedelta(seconds=round(TTEsperaPostoC/NUPostoC,0))))
print('Tempo M dio Espera Fila Posto C (09h-11h): ', str(
    datetime.timedelta(seconds=round(TTEsperaPostoC0911/NUPostoC
    ,0))))
print('Tempo M dio Espera Fila Posto C (11h-13h): ', str(
    datetime.timedelta(seconds=round(TTEsperaPostoC1113/NUPostoC
    ,0))))
print('Tempo M dio Espera Fila Posto C (13h-15h): ', str(
    datetime.timedelta(seconds=round(TTEsperaPostoC1315/NUPostoC
    ,0))))
print('Tempo M dio Espera Fila Posto C (15h-17h): ', str(
    datetime.timedelta(seconds=round(TTEsperaPostoC1517/NUPostoC
    ,0))))
print('Tempo Total Ocupa o Posto C:', str(datetime.timedelta(
    seconds=TTOcupacaoPostoC)))
print('Tempo Total Ocupa o Posto C (09h-11h): ', str(datetime.
    timedelta(seconds=TTOcupacaoPostoC0911)))
print('Tempo Total Ocupa o Posto C (11h-13h): ', str(datetime.
    timedelta(seconds=TTOcupacaoPostoC1113)))
print('Tempo Total Ocupa o Posto C (13h-15h): ', str(datetime.
    timedelta(seconds=TTOcupacaoPostoC1315)))
print('Tempo Total Ocupa o Posto C (15h-17h): ', str(datetime.
    timedelta(seconds=TTOcupacaoPostoC1517)))
print('Total Utentes Tesouraria:', NUTesouraria)
print('Total Utentes Fila Tesouraria:', TUFTesouraria)
print('Tempo M dio Espera Fila Tesouraria:', str(datetime.
    timedelta(seconds=round(TTEsperaTesouraria/NUTesouraria,0))))
print('Tempo M dio Espera Fila Tesouraria (09h-11h): ', str(
    datetime.timedelta(seconds=round(TTEsperaTesouraria0911 /
    NUTesouraria ,0))))
print('Tempo M dio Espera Fila Tesouraria (11h-13h): ', str(
    datetime.timedelta(seconds=round(TTEsperaTesouraria1113 /
    NUTesouraria ,0))))
print('Tempo M dio Espera Fila Tesouraria (13h-15h): ', str(
    datetime.timedelta(seconds=round(TTEsperaTesouraria1315 /
    NUTesouraria ,0))))
print('Tempo M dio Espera Fila Tesouraria (15h-17h): ', str(

```

```
    datetime.timedelta(seconds=round(TTEsperaTesouraria1517 /
    NUTesouraria,0)))
print('Tempo Total Ocupa o Tesouraria:', str(datetime.
    timedelta(seconds=TTOcupacaoTesouraria)))
print('Tempo Total Ocupa o Tesouraria (09h-11h): ', str(
    datetime.timedelta(seconds=TTOcupacaoTesouraria0911)))
print('Tempo Total Ocupa o Tesouraria (11h-13h): ', str(
    datetime.timedelta(seconds=TTOcupacaoTesouraria1113)))
print('Tempo Total Ocupa o Tesouraria (13h-15h): ', str(
    datetime.timedelta(seconds=TTOcupacaoTesouraria1315)))
print('Tempo Total Ocupa o Tesouraria (15h-17h): ', str(
    datetime.timedelta(seconds=TTOcupacaoTesouraria1517)))
print('
    //////////////////////////////////////
')
```

Bibliografia

- [1] Carlos Barrico. *Modelação e Simulação de Sistemas*. Apontamentos do Docente, UBI, 2019.