



Universidade do Minho
Escola de Engenharia

Relatório do Projeto

Licenciatura em Engenharia Biomédica

Algoritmos e Técnicas de Programação
Ano Letivo 2025/2026

Catarina Peixoto, A112402

Joana Sousa, A111697

Mafalda Sequeira, A112113

Docentes: José Carlos Ramalho e Luís Filipe Cunha

7 de janeiro de 2025

Índice

1	Introdução	1
2	Análise e Especificação	2
2.1	Descrição informal do problema	2
2.2	Especificação de requisitos	2
2.2.1	Dados	2
2.2.2	Pedidos	3
3	Conceção da Resolução	5
3.1	Estrutura de dados	5
3.2	Modelação do sistema	8
3.3	Interface gráfica	10
3.3.1	Estrutura geral da Interface	10
3.3.2	Botões de controlo	11
3.3.3	Estrutura de Tabs para resultados	11
4	Codificação e Testes	15
4.1	Alternativas, Decisões e Problemas de Implementação	15
4.1.1	Problemas e Implementação	15
4.2	Exemplo de Execução	17
5	Conclusão	22
6	Referências	23

Lista de Figuras

1	Interface	10
2	Botão Executar simulação	11
3	Botão Sair	11
4	Tab de Log	12
5	Tab de Estatística - Enfermeiros	12
6	Tab de Estatística - Médicos	13
7	Gráfico - Evolução das Filas	13
8	Gráfico - Ocupação dos Enfermeiros	14
9	Gráfico - Ocupação dos Médicos	14
10	Mensagem se inserir texto	15
11	Mensagem se inserir números decimais	16
12	Configurações	18
13	Médicos gerados e as suas especialidades	18
14	Início do Log de simulação	18
15	Taxa de Ocupação dos Enfermeiros	19
16	Taxa de Ocupação dos Médicos	19
17	Gráfico Filas de Espera	20
18	Simulação - Taxa de Ocupação dos Enfermeiros	20
19	Simulação - Taxa de Ocupação dos Médicos	21

1 Introdução

No âmbito da Unidade Curricular de Algoritmos e Técnicas de Programação, foi solicitado o desenvolvimento de uma aplicação em Python para simular o atendimento de doentes numa clínica médica. Este projeto laboratorial, intitulado "Simulação de uma Clínica Médica", tem como objetivo principal a modelação de sistemas do mundo real através de processos probabilísticos, implementando uma simulação de eventos discretos que replica o fluxo de pacientes num ambiente clínico.

A aplicação desenvolvida permite simular a chegada aleatória de doentes segundo uma distribuição de Poisson, o seu atendimento por um conjunto de médicos cujo tempo de consulta segue distribuições probabilísticas (exponencial, normal ou uniforme), e a gestão de uma fila de espera quando todos os médicos se encontram ocupados. Durante a simulação, são registados diversos parâmetros de desempenho, incluindo tempos de espera, tamanhos das filas e percentagem de ocupação dos médicos.

Para além da simulação base, o sistema oferece funcionalidades de análise que permitem estudar o comportamento da clínica sob diferentes condições operacionais, como a variação da taxa de chegada de pacientes ou a alteração dos parâmetros de duração das consultas. A aplicação inclui uma interface gráfica desenvolvida com o módulo simpleGUI, onde é possível configurar parâmetros, executar simulações e visualizar resultados através de gráficos informativos.

O projeto exige ainda a consideração e tratamento de casos específicos do sistema de saúde simulado, o desenvolvimento de uma interface intuitiva, e a capacidade de gerar análises estatísticas relevantes para a gestão de recursos médicos. Através desta implementação, os alunos aplicam conceitos fundamentais de programação, análise de dados e modelação de sistemas, consolidando competências técnicas essenciais para a Engenharia Biomédica.

2 Análise e Especificação

2.1 Descrição informal do problema

O sistema a desenvolver deve simular o funcionamento de uma clínica médica na qual os pacientes chegam de forma aleatória, seguindo uma distribuição de Poisson. Logo a seguir, são triados por enfermeiros, que atribuem pulseiras, também de forma aleatória (verde, amarela, vermelha) indicando a sua prioridade. De seguida, são encaminhados para especialidades médicas específicas onde aguardam em fila de espera até um médico da especialidade estar disponível. Quando este se encontra disponível, os pacientes são atendidos e, no final da consulta, saem da clínica.

O sistema deve registar estatísticas importantes como tempos de espera, tamanhos das filas, taxa de ocupação de médicos e enfermeiros e número de pacientes atendidos. Estes dados seguem distribuições aleatórias, tornando cada simulação diferente e mais próxima da realidade. No final da simulação, o objetivo é analisar estes resultados e perceber como diferentes configurações afetam o desempenho da clínica.

2.2 Especificação de requisitos

2.2.1 Dados

Para a realização deste projeto, foi-nos fornecido um ficheiro JSON (pessoas.json) contendo informações de pessoas simuladas que serão utilizadas como pacientes em formato de dicionário. Cada pessoa possui:

- Id: identificador único (formato "p1", "p2", etc.)
- Nome: nome completo do paciente
- Idade: idade do paciente
- Sexo: sexo do paciente
- Informações adicionais conforme necessário

Para além disso, foi também fornecido pelos docentes um código base em Python (`clinic_alunos.py`) que implementa uma versão simplificada da simulação, servindo como ponto de partida para o desenvolvimento do projeto completo.

O código base implementa um cenário mais simples onde os pacientes chegam aleatoriamente à clínica segundo uma distribuição de Poisson; são atendidos por médicos disponíveis em regime de "primeiro a chegar, primeiro a ser atendido"; se não há médico disponível, aguardam numa fila de espera; o tempo de consulta varia de acordo com uma distribuição configurável (exponencial, normal ou uniforme); e após a consulta, o paciente sai e o médico fica disponível para atender o próximo da fila.

Após a análise cuidadosa deste código, fomos capazes de colocá-lo noutra patamar, implementando mais funções que vão ser descritas já de seguida.

2.2.2 Pedidos

Como vimos antes, são pedidas diversas funções na formulação do sistema, em seguida enumeramos o que é pedido para cada uma das funções requisitadas.

Parâmetros de configuração

Portanto, deve haver no trabalho, parâmetros configuráveis como:

- Número de enfermeiros na clínica
- Número de médicos disponíveis
- Taxa de chegada de pacientes (pacientes/hora)
- Tempo médio de triagem (minutos)
- Tempo médio de consulta (minutos)
- Duração total da simulação (minutos)
- Tipo de distribuição (exponencial, normal, uniform)

Resultados esperados

Tal como qualquer trabalho, é esperado atingirmos resultados específicos, neste caso é fundamental reconhecer:

- Tempo médio de espera para os doentes
- Tempo médio de consulta
- Quanto tempo em média esteve cada doente na clínica
- Tamanho da fila de espera (média e máximo)
- Percentagem de tempo em que os médicos estão ocupados
- Número total de doentes atendidos

Interface gráfica

Deverá ter uma interface gráfica desenvolvida com o módulo simpleGUI na qual deverá ser possível:

1. **Executar uma simulação** - Processar eventos de chegada, triagem e saída.
2. **Alterar os parametros da simulação** - Permitir ao utilizador definir todos os parâmetros antes de executar a simulação.
3. **Visualizar resultados** - Tabela de Log que mostra cronologicamente todos os eventos da simulação.
4. **Visualização de gráficos** - a fila de espera e ocupação dos médicos.

3 Conceção da Resolução

3.1 Estrutura de dados

Para a realização deste projeto, foi necessário definir cuidadosamente as estruturas de dados que suportariam toda a lógica da simulação. A escolha adequada destas estruturas foi crucial para garantir eficiência, clareza do código e facilidade de manutenção.

Fila de Eventos

A simulação baseia-se no conceito de eventos discretos, organizados numa lista cronológica denominada `queueEventos`. Cada evento é representado por um tuplo de três elementos: o tempo de ocorrência (valor decimal em minutos), o tipo de evento ("`chegada`", "`fim_triagem`" ou "`saída`") e o id do paciente envolvido.

Para garantir o funcionamento correto, a lista deve manter sempre a ordem temporal. Assim, ao criar um novo evento, é preciso inseri-lo na posição adequada. Para isso existem três funções: uma que encontra a posição de inserção, outra que adiciona o evento (`enqueue`) e uma terceira que remove o próximo evento a processar (`dequeue`).

Fila de espera

O sistema possui duas filas de espera distintas, cada uma servindo um propósito específico no fluxo de atendimento.

A primeira é a fila de triagem (`fila_triagem`), onde aguardam os pacientes que chegaram à clínica mas ainda não foram atendidos por um enfermeiro. Cada elemento desta fila é um tuplo contendo o id do paciente e o momento em que chegou à clínica.

A segunda é a fila de médicos (`queue`), significativamente mais complexa que a anterior. Aqui aguardam os pacientes que já foram triados e estão à espera de serem atendidos por um médico da especialidade apropriada. Esta fila não funciona num simples regime de "primeiro a chegar, primeiro a ser atendido" - em vez disso, implementa um sistema de prioridades baseado na cor da pulseira atribuída durante a triagem. Quando um médico fica disponível, o sistema procura na fila o paciente com maior prioridade (pulseira vermelha > amarela > verde) que necessita daquela especialidade específica. Em caso de empate na prioridade, prevalece quem chegou primeiro.

Dicionário do Paciente

Toda a informação sobre cada paciente é armazenada num dicionário Python (**doentes**), onde a chave é o id único do paciente e o valor é outro dicionário contendo todos os seus dados. Estes dados incluem as informações originais carregadas do ficheiro JSON (como nome e ID), bem como atributos atribuídos dinamicamente durante a simulação que seguem a seguinte distribuição:

Cor da pulseira:

- Verde: 70% dos casos (baixa prioridade)
- Amarela: 20% dos casos (prioridade média)
- Vermelha: 10% dos casos (alta prioridade)

Especialidade médica:

- Medicina Geral: 35%
- Ortopedia: 20%
- Gastroenterologia: 15%
- Cardiologia: 15%
- Neurologia: 10%
- Psiquiatria: 5%

Lista de Enfermeiros

Os enfermeiros são representados numa lista (**enfermeiros**), onde cada enfermeiro é outra lista contendo cinco elementos: identificador (string no formato "e1", "e2", etc.), estado de ocupação (booleano indicando se está a atender alguém), identificador do paciente atual (None se estiver livre), tempo total acumulado em que esteve ocupado (para calcular estatísticas) e o momento em que iniciou a triagem atual.

Esta estrutura permite não apenas saber quem está disponível, mas também calcular a taxa de ocupação de cada enfermeiro ao longo da simulação. Para equilibrar a carga de trabalho, implementamos um algoritmo que, ao procurar um enfermeiro disponível, seleciona aquele que tem o menor tempo total de ocupação acumulado.

Lista de médicos

Semelhante aos enfermeiros, os médicos são representados numa lista (`medicos`), mas com um elemento adicional: a especialidade. Cada médico é uma lista com seis elementos: identificador, estado de ocupação, paciente atual, tempo total ocupado, momento de início da consulta atual e especialidade médica.

A atribuição de especialidades aos médicos é feita durante a inicialização da simulação, seguindo uma distribuição específica:

- Medicina Geral: 30%
- Ortopedia: 20%
- Gastroenterologia: 15%
- Cardiologia: 20%
- Neurologia: 10%
- Psiquiatria: 5%

Os primeiros médicos criados recebem automaticamente as seis especialidades principais (garantindo que há pelo menos um médico de cada especialidade), e os restantes recebem especialidades de forma probabilística, com maior peso para Medicina Geral e Cardiologia.

Estruturas para visualização gráfica

Para permitir a criação dos gráficos analíticos, implementamos cinco listas paralelas que registam a evolução do sistema ao longo do tempo:

- `h_t`: regista cada momento de tempo em que ocorreu um evento
- `h_f_tri`: regista o tamanho da fila de triagem em cada momento
- `h_f_med`: regista o tamanho da fila de médicos em cada momento
- `h_oc_med`: regista a percentagem de médicos ocupados em cada momento
- `h_oc_enf`: regista a percentagem de enfermeiros ocupados em cada momento

Estas listas são atualizadas a cada evento processado, criando um histórico completo que alimenta os gráficos apresentados na interface.

3.2 Modelação do sistema

A modelação do sistema foi concebida para replicar de forma realista o funcionamento de uma clínica médica, capturando a complexidade das interações entre pacientes, enfermeiros e médicos.

Fluxo do atendimento

O ciclo de vida de um paciente na clínica passa por três fases principais.

Na primeira fase, um paciente chega à clínica. Este momento é um evento de tipo "**chegada**" gerado antes do início da simulação, seguindo uma distribuição de Poisson que modela chegadas aleatórias mas com uma taxa média definida. Quando este evento é processado, o sistema carrega os dados do paciente tenta atribuí-lo a um enfermeiro disponível. Se existe um enfermeiro livre (especificamente, aquele que tem o menor tempo acumulado de trabalho), a triagem inicia imediatamente e é agendado um evento de "**fim_triagem**" para o futuro. Caso contrário, o paciente entra na fila de triagem e aguarda.

A segunda fase inicia quando ocorre um evento de "**fim_triagem**". Neste momento, o enfermeiro que realizou a triagem fica disponível novamente e, se há pacientes na fila de triagem, o próximo é imediatamente chamado. O paciente triado recebe então dois atributos cruciais: uma cor de pulseira e uma especialidade médica necessária (determinadas aleatoriamente de acordo com as distribuições mencionadas em 3.1). Logo a seguir, o sistema procura um médico disponível da especialidade correta e, se encontrado, a consulta inicia e agenda-se um evento de "**saída**". Se não, o paciente entra na fila de médicos.

A terceira e última fase ocorre com o evento de "**saída**", quando a consulta termina. O médico fica livre e incrementa-se o contador de pacientes atendidos. Crucialmente, se existe uma fila de espera, não é simplesmente chamado o próximo paciente, em vez disso, executa-se um algoritmo de priorização que percorre toda a fila procurando o paciente prioritário que necessita daquela especialidade específica.

Sistema de prioridades

O sistema de priorização implementado é sofisticado e multi-critério. Quando um médico de uma determinada especialidade fica disponível, o algoritmo percorre a fila de espera considerando três critérios hierárquicos:

1. Considera apenas pacientes que necessitam daquela especialidade específica (exemplo: um paciente que precisa de Ortopedia nunca será atendido por um cardiologista, independentemente da sua prioridade).
2. Entre os pacientes da especialidade correta, dá preferência absoluta à cor da pulseira: vermelha (valor 1) é atendida antes de amarela (valor 2), que é atendida antes de verde (valor 3).
3. Em caso de empate na cor da pulseira, prevalece quem chegou primeiro (FIFO - First In, First Out).

Distribuições Probabilísticas

A simulação utiliza três distribuições probabilísticas diferentes, cada uma modelando um aspecto específico do sistema:

- **Distribuição de Poisson** governa as chegadas de pacientes. Esta é uma escolha clássica em teoria das filas porque modela bem eventos aleatórios que ocorrem de forma independente a uma taxa média constante.
- **Distribuição Exponencial** é usada por defeito para os tempos de triagem e pode também ser usada para os tempos de consulta.
- **Distribuição normal** para os tempos de consulta, o que gera valores concentrados em torno da média com menor variabilidade, ou uma distribuição uniforme, onde qualquer valor entre metade e uma vez e meia da média é igualmente provável.

3.3 Interface gráfica

A interface gráfica foi desenvolvida com o objetivo de tornar a simulação acessível, intuitiva e visualmente informativa, permitindo que utilizadores sem conhecimentos de programação possam explorar diferentes cenários.

3.3.1 Estrutura geral da Interface

A janela principal foi organizada seguindo princípios de design de interfaces (1). No topo, um título em destaque identifica a aplicação. Abaixo, encontra-se a secção de configurações, onde todos os parâmetros da simulação podem ser ajustados através de campos de entrada numéricos simples. Esta secção inclui:

- Número de enfermeiros (valor inteiro)
- Número de médicos (valor inteiro)
- Taxa de chegada de pacientes em pacientes por hora (valor inteiro)
- Duração total da simulação em minutos (valor inteiro)
- Tempo médio de triagem em minutos (valor decimal)
- Tempo médio de consulta em minutos (valor decimal)
- Tipo de distribuição para tempos de consulta (menu de seleção)

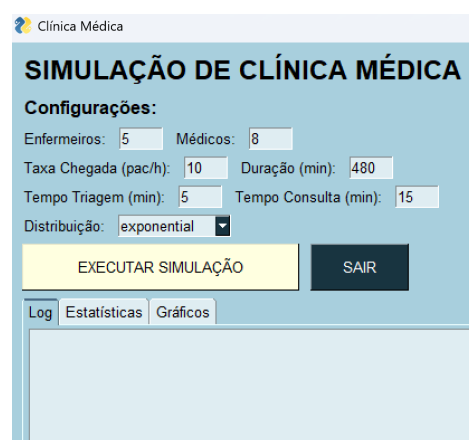


Figura 1: Interface

3.3.2 Botões de controlo

Dois botões principais controlam a aplicação, o botão "Executar Simulação" e o botão Sair.

Um aspeto importante do design foi garantir que a simulação pode ser executada múltiplas vezes sem fechar a aplicação. Cada execução limpa os resultados anteriores e processa uma nova simulação independente, permitindo comparações rápidas entre diferentes configurações.

Botão Executar Simulação

O botão "Executar Simulação"(2), destacado visualmente com cor de fundo amarela clara, inicia todo o processo de simulação com os parâmetros atualmente configurados.

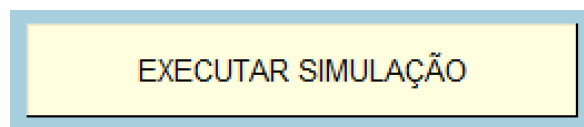


Figura 2: Botão Executar simulação

Botão Sair

O botão "SAIR"(3) permite fechar a aplicação de forma limpa.

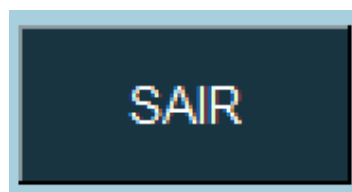


Figura 3: Botão Sair

3.3.3 Estrutura de Tabs para resultados

Os resultados são apresentados através de um sistema de tabs (separadores) que organiza a informação em três categorias distintas:

Tab de Log

A Tab de Log (4) apresenta um registo cronológico completo de todos os eventos da simulação. Cada linha mostra o tipo de evento (Chegada, Triagem, Fim Triagem, Atendido, Espera, Saída), o tempo em que ocorreu, o id do paciente envolvido e detalhes adicionais contextuais. Por exemplo, ao ver "Triado", são mostradas a cor da pulseira atribuída e a especialidade. Este log é apresentado numa caixa de texto de múltiplas linhas, não editável mas com scroll automático.

Log	Estatísticas	Gráficos
TRIADO	91.71	p13 Triado: amarela Gastroenterologia
ATENDIDO	91.71	p13 Atendido por m3 (Gastroenterologia)
FIM TRIAGEM	92.63	p11 Terminou triagem
TRIADO	92.63	p11 Triado: amarela Ortopedia
ATENDIDO	92.63	p11 Atendido por m8 (Ortopedia)
CHEGADA	94.46	p14 Entrou na clínica
TRIAGEM	94.46	p14 Iniciou triagem com e4
SAÍDA	94.75	p13 Saiu da clínica (Gastroenterologia)
SAÍDA	94.97	p11 Saiu da clínica (Ortopedia)
FIM TRIAGEM	95.32	p14 Terminou triagem

Figura 4: Tab de Log

Tab de Estatística

A Tab de Estatísticas apresenta métricas calculadas sobre o desempenho do sistema. Para cada enfermeiro, mostra o seu identificador e a percentagem de tempo que passou ocupado durante toda a simulação (5). Para cada médico, mostra adicionalmente a sua especialidade entre parênteses. Estas estatísticas permitem identificar rapidamente se há recursos subutilizados (baixa ocupação) ou sobrecarregados (ocupação próxima de 100%)(6).

Log	Estatísticas	Gráficos
ESTATÍSTICAS ENFERMEIROS:		
e1:	Ocupação	10.34%
e2:	Ocupação	10.99%
e3:	Ocupação	10.33%
e4:	Ocupação	10.41%
e5:	Ocupação	13.57%

Figura 5: Tab de Estatística - Enfermeiros

Log	Estatísticas	Gráficos
ESTATÍSTICAS MÉDICOS:		
m1	(Medicina Geral):	69.94%
m2	(Ortopedia):	48.63%
m3	(Gastroenterologia):	47.98%
m4	(Cardiologia):	29.61%
m5	(Neurologia):	7.77%
m6	(Psiquiatria):	12.35%
m7	(Psiquiatria):	10.80%
m8	(Ortopedia):	9.36%

Figura 6: Tab de Estatística - Médicos

Tab de Gráficos

A Tab de Gráficos é a mais complexa visualmente. Apresenta três gráficos de linha empilhados verticalmente, cada um mostrando a evolução temporal de métricas diferentes:

- O primeiro gráfico mostra simultaneamente duas linhas: o tamanho da fila de triagem (azul turquesa) e o tamanho da fila de médicos (rosa choque), permitindo comparar visualmente como as duas filas evoluem (7)

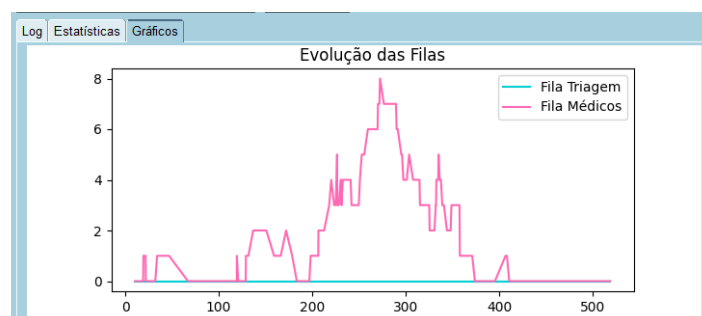


Figura 7: Gráfico - Evolução das Filas

- O segundo gráfico mostra a taxa de ocupação dos enfermeiros ao longo do tempo, em percentagem (8)

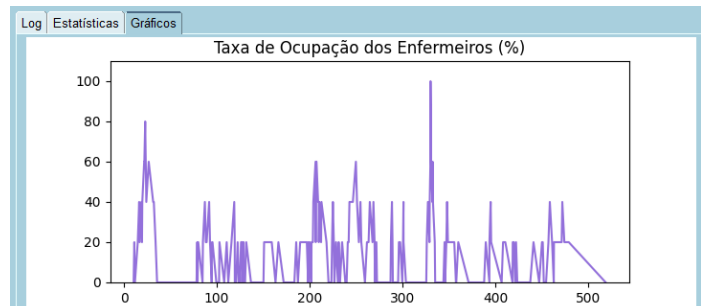


Figura 8: Gráfico - Ocupação dos Enfermeiros

- O terceiro gráfico mostra a taxa de ocupação dos médicos ao longo do tempo, também em percentagem (9)

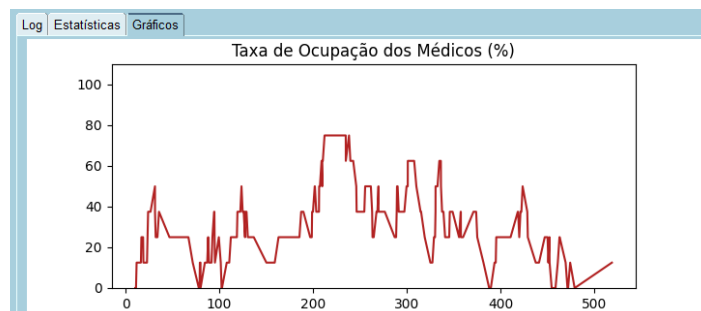


Figura 9: Gráfico - Ocupação dos Médicos

Esta secção foi implementada como uma coluna com scroll vertical, permitindo que o utilizador visualize confortavelmente os três gráficos mesmo em ecrãs mais pequenos.

4 Codificação e Testes

4.1 Alternativas, Decisões e Problemas de Implementação

Durante o desenvolvimento deste projeto, foram encontrados diversos desafios técnicos que exigiram análise cuidadosa, pesquisa de soluções e, em alguns casos, refatorização significativa do código. Esta seção documenta os principais problemas enfrentados e as soluções implementadas.

4.1.1 Problemas e Implementação

Validação de Inputs Numéricos

Um dos primeiros problemas identificados relacionou-se com a entrada de dados por parte do utilizador. Como a interface permite configurar diversos parâmetros numéricos, era essencial garantir que apenas valores válidos fossem aceites.

A solução implementada envolveu uma validação em duas etapas. Primeiro, todos os valores são convertidos para decimais (float) dentro de um bloco try-except, capturando qualquer tentativa de inserir texto não numérico. Se esta conversão for bem-sucedida, verifica-se então quais campos devem ser especificamente inteiros (número de enfermeiros, número de médicos, taxa de chegada e duração da simulação). Apenas se ambas as validações passarem é que os valores são aceites e a simulação pode prosseguir.

Esta abordagem permite dar feedback específico ao utilizador: se inserir texto, recebe uma mensagem a indicar que deve introduzir apenas números (10); se inserir um decimal num campo que requer inteiro, recebe uma mensagem diferente explicando que aqueles campos específicos devem ser números inteiros (11).

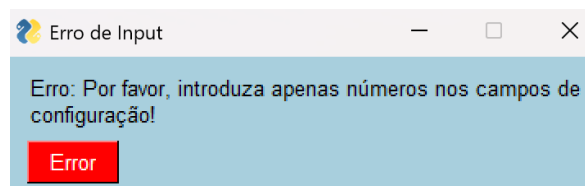


Figura 10: Mensagem se inserir texto

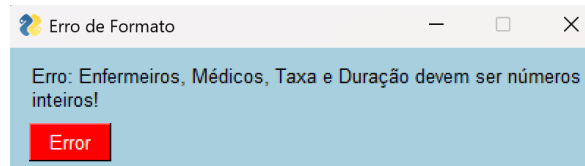


Figura 11: Mensagem se inserir números decimais

Gestão de múltiplos Gráficos

A visualização gráfica apresentou um problema particular: quando a simulação era executada múltiplas vezes, os novos gráficos sobrepunham-se aos anteriores em vez de os substituir, criando uma interface confusa e consumindo memória progressivamente.

A solução implementada verifica, antes de criar novos gráficos, se já existem canvases anteriores. Se existirem, os seus widgets Tkinter são explicitamente destruídos. Este processo é repetido para os três canvases (fila de espera, ocupação de enfermeiros e ocupação de médicos). Apenas após garantir que todos os canvases anteriores foram destruídos é que se criam as novas figuras matplotlib e se desenham os novos gráficos.

Balanco de carga entre Enfermeiros

Durante os testes iniciais, observou-se um padrão não desejado: o primeiro enfermeiro da lista tinha consistentemente uma taxa de ocupação muito superior aos restantes. A análise do código revelou que a função de procura de enfermeiro livre simplesmente retornava o primeiro que encontrasse disponível, sem qualquer critério de balanceamento.

Para resolver este problema, foi implementado um algoritmo que, em vez de retornar o primeiro enfermeiro livre, percorre todos os enfermeiros disponíveis e seleciona aquele que tem o menor tempo total acumulado de ocupação. Isto distribui a carga de trabalho equitativamente.

Sistema de prioridades na fila de Médicos

A implementação do sistema de prioridades foi um dos pontos mais difíceis, porque não bastava ordenar a fila por cor da pulseira: era preciso considerar especialidade, prioridade e tempo de chegada ao mesmo tempo.

O algoritmo faz uma procura linear, mantendo o melhor candidato encontrado até então. Para isso, começa com valores sentinela: índice = -1, prioridade = 999 e tempo de chegada = 999999.

Depois percorre toda a fila. Para cada paciente, verifica primeiro se a especialidade necessária coincide com a do médico livre; se não coincidir, ignora-o. Se coincidir, obtém a prioridade numérica da pulseira (vermelha=1, amarela=2, verde=3).

Um paciente passa a ser o novo “melhor” se tiver prioridade superior (número menor) ou, em caso de empate, se tiver chegado mais cedo.

No fim, se o índice encontrado não for -1, remove-se esse paciente da fila, mesmo que não esteja na primeira posição, e inicia-se o seu atendimento.

4.2 Exemplo de Execução

Para demonstrar o funcionamento do sistema, apresentamos um cenário de teste que ilustra o funcionamento equilibrado da simulação, onde os recursos são dimensionados adequadamente para a demanda esperada (12). Os parâmetros utilizados foram:

- Enfermeiros: 5
- Médicos: 8
- Taxa de Chegada: 10 pacientes/hora
- Tempo Médio de Triagem: 5 minutos
- Tempo Médio de Consulta: 15 minutos
- Duração da Simulação: 480 minutos (8 horas de funcionamento)
- Distribuição dos Tempos de Consulta: Exponencial

Configurações:

Enfermeiros: Médicos:

Taxa Chegada (pac/h): Duração (min):

Tempo Triagem (min): Tempo Consulta (min):

Distribuição:

Figura 12: Configurações

Ao executar a simulação com estes parâmetros, o sistema primeiro apresentou no terminal a tabela de especialidades atribuídas aos médicos. Esta distribuição é feita automaticamente pelo sistema, garantindo que todas as seis especialidades têm pelo menos um médico:

Log	Estatísticas	Gráficos
Médico	Especialidade	
m1	Medicina Geral	
m2	Ortopedia	
m3	Gastroenterologia	
m4	Cardiologia	
m5	Neurologia	
m6	Psiquiatria	
m7	Gastroenterologia	
m8	Neurologia	

Figura 13: Médicos gerados e as suas especialidades

O log gerado apresentou cronologicamente todos os eventos da simulação (14).

Log	Estatísticas	Gráficos	
Evento	Tempo	Doente	Detalhes
CHEGADA	1.66	p1	Entrou na clínica
TRIAGEM	1.66	p1	Iniciou triagem com e1
FIM TRIAGEM	5.74	p1	Terminou triagem
TRIADO	5.74	p1	Triado: verde Gastroenterologia
ATENDIDO	5.74	p1	Atendido por m3 (Gastroenterologia)
SAÍDA	8.35	p1	Saiu da clínica (Gastroenterologia)
CHEGADA	10.35	p2	Entrou na clínica
TRIAGEM	10.35	p2	Iniciou triagem com e2

Figura 14: Início do Log de simulação

As estatísticas calculadas ao final da simulação mostraram um sistema bem balanceado. Para os enfermeiros, as taxas de ocupação foram:

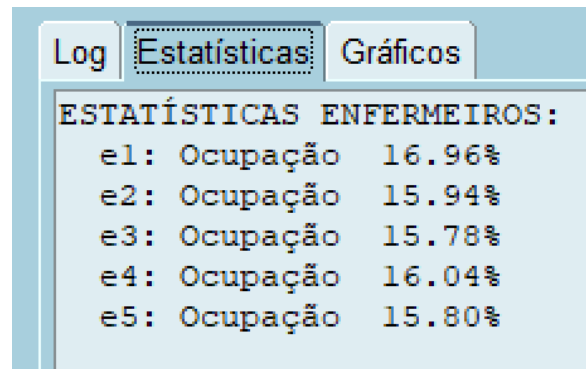


Figura 15: Taxa de Ocupação dos Enfermeiros

A proximidade destes valores confirma que o algoritmo de balanceamento de carga está a funcionar corretamente. Nenhum enfermeiro ficou significativamente mais sobrecarregado do que os restantes.

Para os médicos, as taxas de ocupação variaram mais significativamente devido à variação das diferentes probabilidades de cada especialidade:

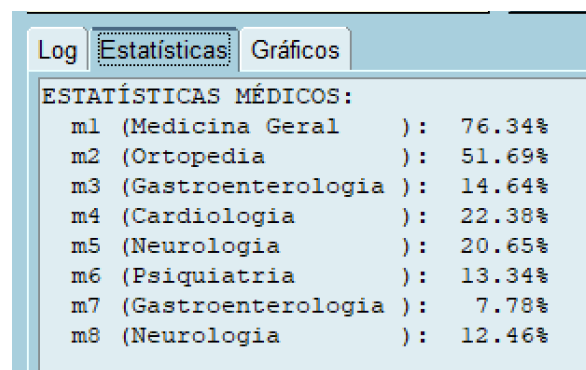


Figura 16: Taxa de Ocupação dos Médicos

O primeiro gráfico, (17) que mostra a evolução das filas, revelou padrões interessantes. A fila de triagem manteve-se uma linha reta, revelando que 5 enfermeiros chega perfeitamente para a taxa de chegada configurada.

A fila de médicos, em contraste, apresentou um comportamento mais irregular. Iniciou próxima de zero, cresceu gradualmente nas primeiras 2 horas até um pico de cerca de 14 pacientes por volta dos 300 minutos, depois foi diminuindo até ao fim da simulação.

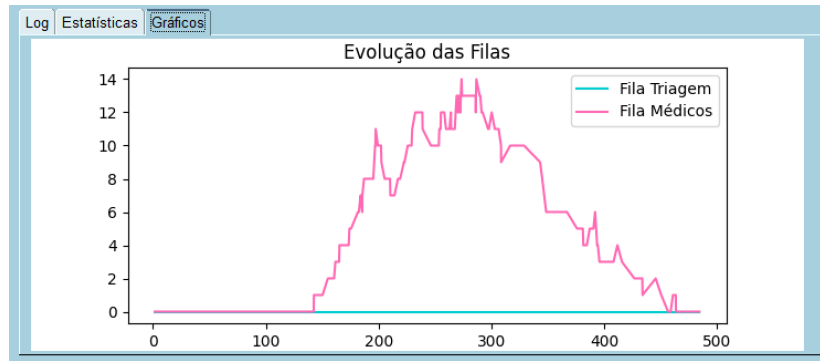


Figura 17: Gráfico Filas de Espera

O segundo gráfico (18) mostrou a taxa de ocupação dos enfermeiros oscilando entre 20% e 80%. Os períodos de 100% correspondiam a momentos onde todos os enfermeiros estavam simultaneamente ocupados. A média visual situou-se em torno de 20%, consistente com as estatísticas calculadas.

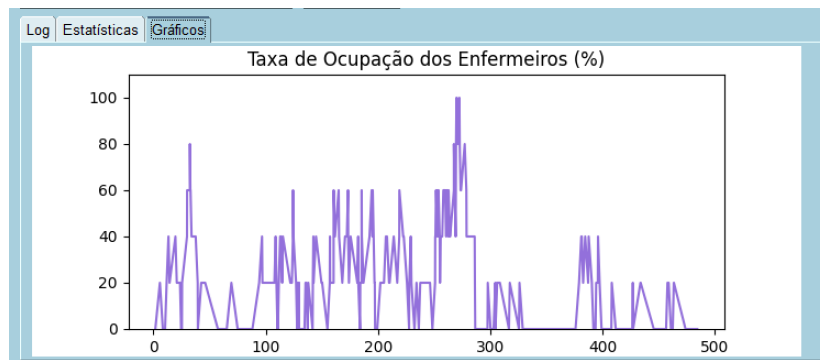


Figura 18: Simulação - Taxa de Ocupação dos Enfermeiros

O terceiro gráfico (19), de ocupação dos médicos, apresentou maior variabilidade. A taxa oscilou entre 20% e 80%, refletindo a heterogeneidade das especialidades. Períodos de alta ocupação ($>90\%$) indicavam momentos onde quase todos os médicos estavam ocupados, o que, neste caso, acabou por não acontecer. Enquanto os períodos de menor ocupação refletiam desbalanceamentos temporários entre oferta e demanda de especialidades específicas.

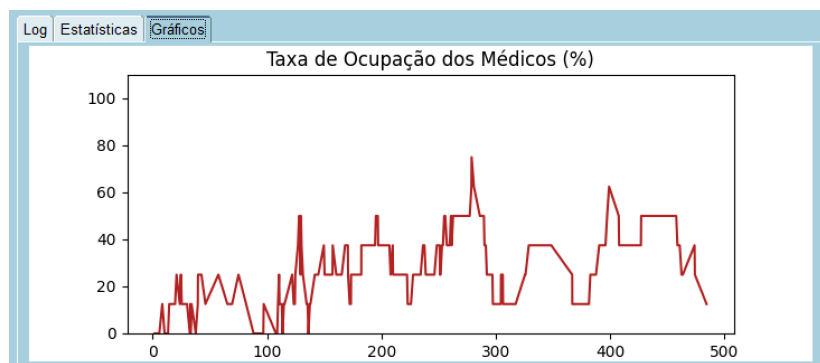


Figura 19: Simulação - Taxa de Ocupação dos Médicos

5 Conclusão

O desenvolvimento deste projeto revelou-se extremamente enriquecedor. A complexidade de coordenar múltiplos recursos (enfermeiros e médicos), gerir filas com diferentes prioridades e simular comportamentos estocásticos exigiu raciocínio lógico avançado e atenção aos detalhes.

Em suma, o projeto cumpriu os objetivos propostos e proporcionou uma sólida base de conhecimentos em simulação de sistemas, programação Python e análise de dados, competências essenciais para a Engenharia Biomédica.

6 Referências

Ficheiros fornecidos pelos docentes para realização do projeto

<https://ep1.di.uminho.pt/~jcr/AULAS/ATP2025/aulas2025.html>

<https://pythontutor.com/>