Benchmark de Estruturas de Fila com Prioridade

# Objetivo

Comparar duas implementações de estrutura de fila com prioridade, considerando os seguintes critérios:  
- Desempenho  
- Eficiência  
- Complexidade  
- Tamanho do código

# Estruturas Comparadas

|  |  |
| --- | --- |
| Estrutura | Descrição |
| Código Proposto (Custom) | Implementação com OOP, nós personalizados e lógica de triagem hospitalar |
| Código Alternativo (ChatGPT) | Fila com prioridade simples baseada em lista encadeada |

# Análise Técnica

|  |  |  |
| --- | --- | --- |
| Critério | Código ChatGPT | Código Proposto |
| Desempenho (teórico) | Boa performance para cenários pequenos. Inserção O(n), remoção O(1) | Inserção O(n), mas com mais verificações. Remoção O(1) |
| Eficiência de memória | Baixo consumo, estrutura simples | Maior uso de memória por atributos adicionais e objetos |
| Complexidade de implementação | Baixa. Estrutura linear simples | Alta. Uso de OOP, herança, múltiplos atributos |
| Flexibilidade | Baixa. Fixado a tipos e lógica específicos | Alta. Reutilizável com generics e lógica médica |
| Tamanho do código | 53 linhas / 1510 bytes | 129 linhas / 4747 bytes |

# Interpretação

O código ChatGPT é eficiente em termos de tamanho e simplicidade, adequado para testes rápidos ou simulações simples.  
O código proposto foi desenvolvido para refletir um ambiente real (pronto-socorro), com prioridades clínicas, tempo de atendimento e máxima urgência.  
Apesar de ser mais pesado, o design orientado a objetos e extensibilidade o tornam mais adequado para integrações futuras com sistemas de banco de dados e interface gráfica, como exigido nas próximas fases da prova.

# Sugestão de Teste Prático

Inclua um teste com 1000 inserções e 1000 remoções em ambos os códigos. Você pode usar:  
  
console.time("enqueue");  
for (...) { fila.enqueue(...) }  
console.timeEnd("enqueue");

# Conclusão

O código proposto é mais robusto e alinhado ao objetivo da aplicação hospitalar. A comparação mostra que, embora menos performático em termos de tamanho e simplicidade, ele é mais adequado para uso em sistemas reais, garantindo flexibilidade, organização e expansão futura — características desejáveis para aplicações em produção.