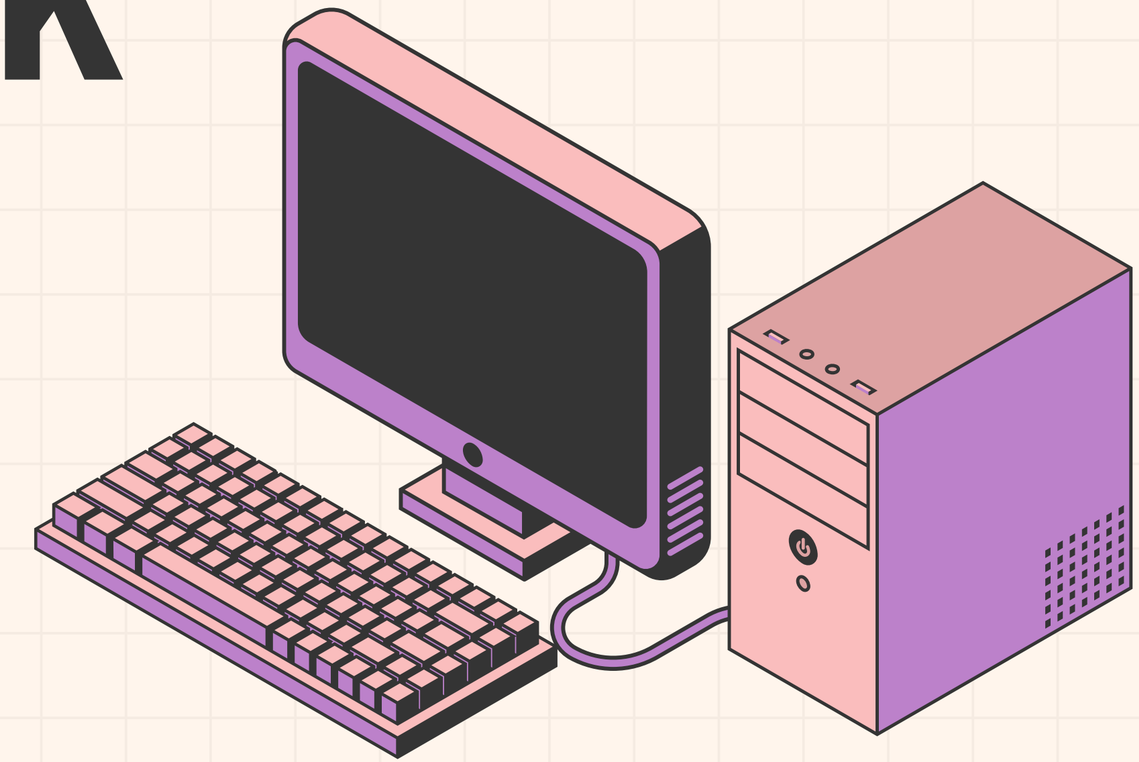
BENCHMARK

DESEMPENHO

&

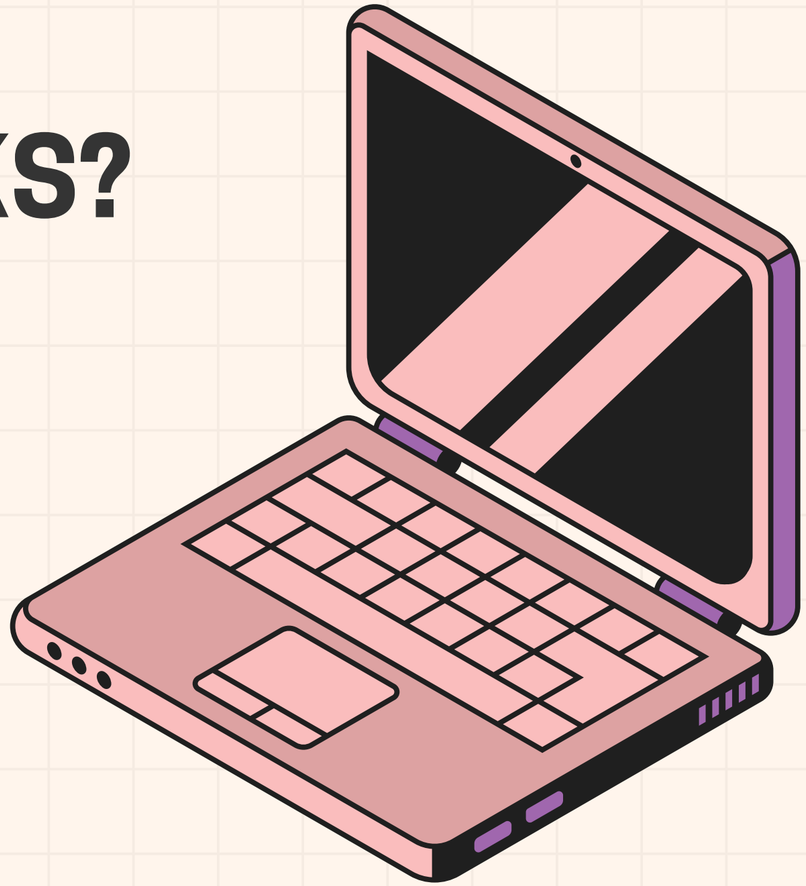
CORRETUDE



Alunos	Matricula
Luiz Henrique Botega Beraldi	202200560378
Murilo Aldigueri Marino	202200560390

O QUE SÃO BENCHMARKS?

Benchmarks são testes ou conjuntos de testes usados para avaliar o desempenho, eficiência ou corretude de sistemas, hardware, software ou algoritmos, eles fornecem métricas quantificáveis para comparação



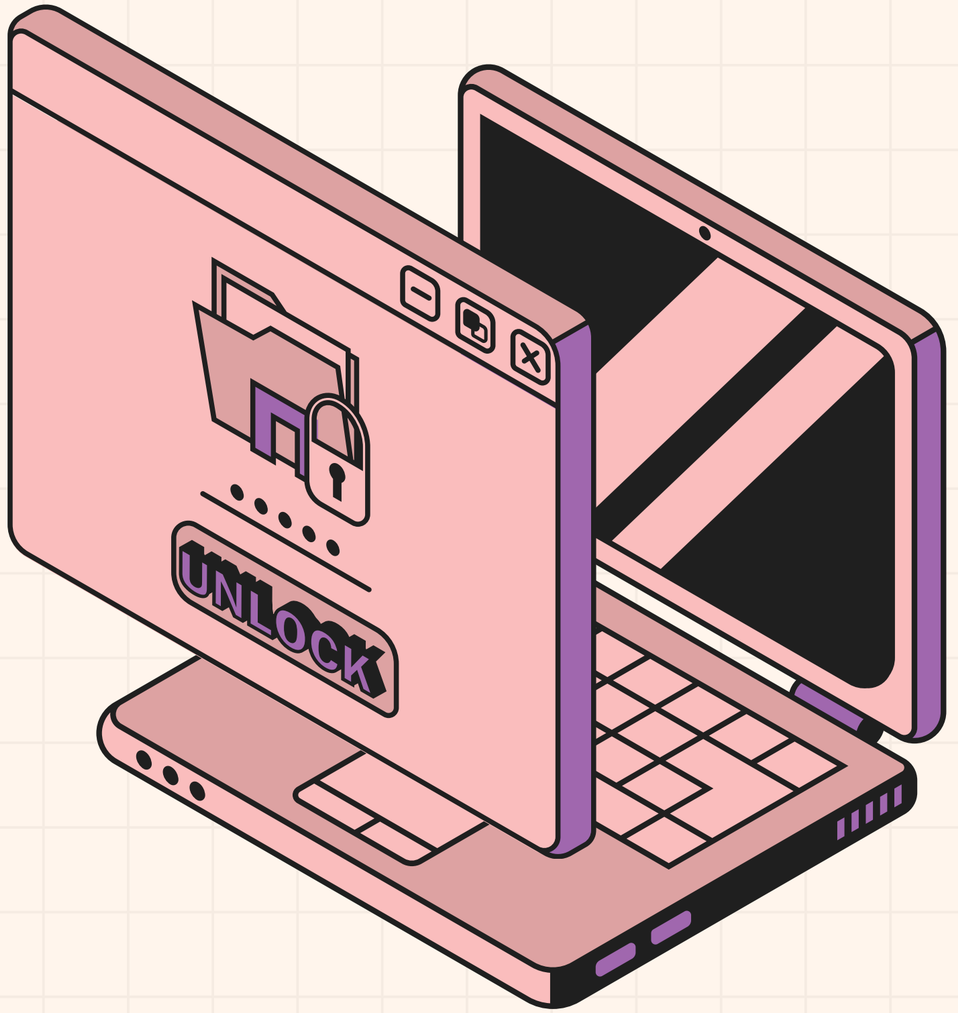
QUANDO SURGIRAM?

Anos 50: Surgiram os primeiros benchmarks para comparar o desempenho de sistemas como IBM 704 e 709, focando em operações básicas como cálculos matemáticos

FORTRAN: Benchmarks passaram a avaliar a eficiência de compiladores e sistemas em cálculos científicos

Anos 70: Benchmarks mais formais(Whetstone e Dhrystone)

Anos 80: Criação do SPEC, padrão da indústria para avaliar desempenho de CPUs e sistemas completos



CONCEITOS

OBJETIVOS

Desempenho: Medir velocidade, eficiência energética, escalabilidade, etc

Corretude: Verificar se um sistema ou algoritmo funciona conforme o esperado

CONTEXTO

Hardware: Comparação de processadores, GPUs, discos rígidos

Software: Testes de aplicativos, sistemas operacionais, bancos de dados

Algoritmos: Avaliação de eficiência computacional

IMPORTÂNCIA

Para desenvolvedores: Identificar gargalos e otimizar sistemas

Para consumidores: Escolher produtos com base em dados objetivos

Para pesquisadores: Validar novas abordagens e compará-las

COMO FUNCIONAM OS BENCHMARKS?

Configuração

O usuário define os parâmetros do teste

Execução

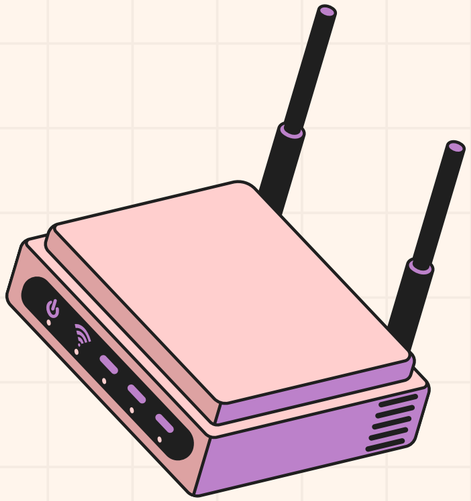
A ferramenta executa os testes de forma automatizada, simulando cenários **reais** ou **sintéticos**

Coleta de dados

Métricas como tempo de execução, uso de recursos e precisão são coletadas

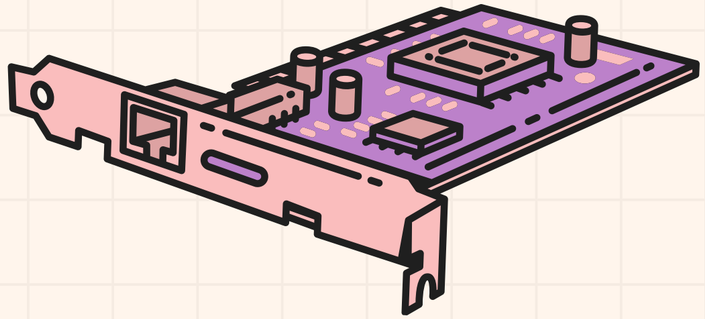
Análise

Os resultados são comparados com benchmarks anteriores ou com sistemas concorrentes



Sintéticas: Criadas artificialmente para medir características específicas, como a velocidade de cálculos em CPU

Reais: Baseadas em aplicações do mundo real, como rodar uma simulação de jogo ou renderizar um vídeo



FERRAMENTAS & FRAMEWORKS

DESEMPENHO

CPU/GPU

3DMark

Armazenamento

CrystalDiskMark

Bancos de Dados

HammerDB

CORRETUDE

Testes de Software

Pyteste

Testes de Software

JUnit

COMPARAÇÃO ENTRE BENCHMARKS
LUIZ VS MURILO

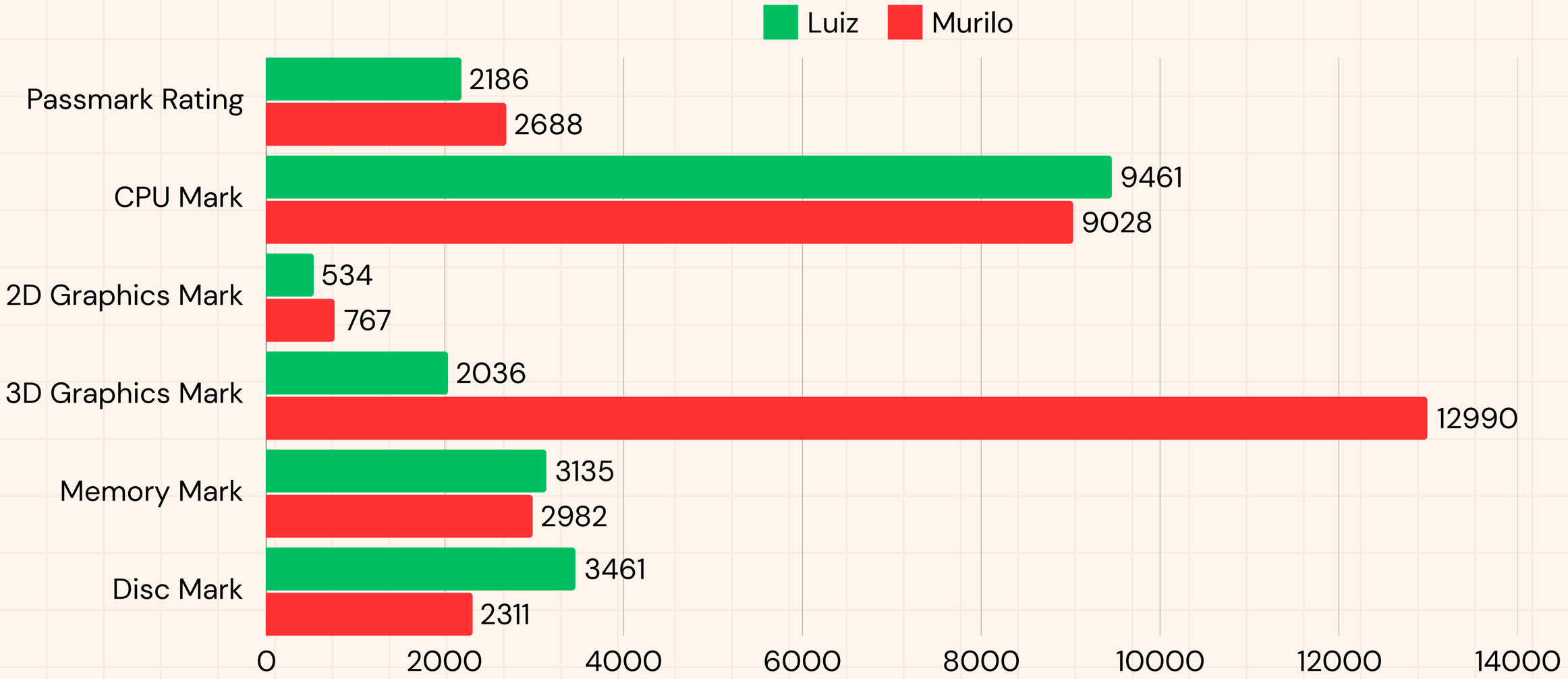
ESPECIFICAÇÕES

PROCESSADOR: INTEL CORE I3-10100F @ 3.60GHZ
PLACA DE VÍDEO: NVIDIA GTX 550 TI
MEM. RAM: 2 X 8RAM
DISCO RÍGIDO: SSD WD 480GB

ESPECIFICAÇÕES

PROCESSADOR: INTEL CORE I5-9400F @ 2.90GHZ
PLACA DE VÍDEO: NVIDIA GTX 1660 SUPER
MEM. RAM: 2 X 8RAM
DISCO RÍGIDO: SSD WD 480GB

COMPARAÇÃO ENTRE BENCHMARKS



*TESTES FEITOS NO PASSMARK SOFTWARE

BENCHMARKS NA INDÚSTRIA

TECNOLOGIA

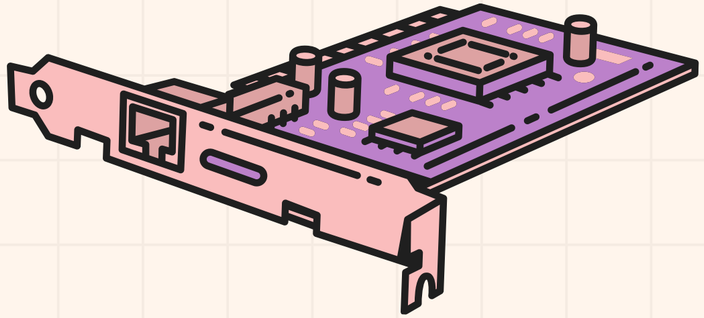
Google, Amazon e Microsoft usam benchmarks para otimizar servidores, sistemas de nuvem e algoritmos.

PESQUISA CIENTÍFICA

São usados para comparar algoritmos de machine learning, simulações físicas e processamento de grandes volumes de dados.

JOGOS

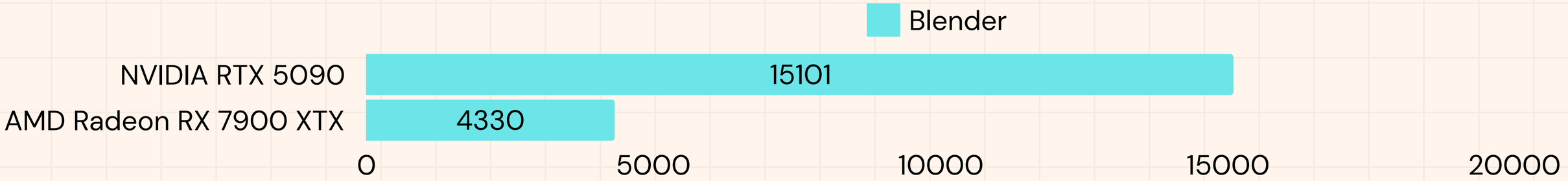
NVIDIA e AMD usam benchmarks para testar placas de vídeo em jogos pesados (ex: Cyberpunk 2077, Red Dead Redemption 2).

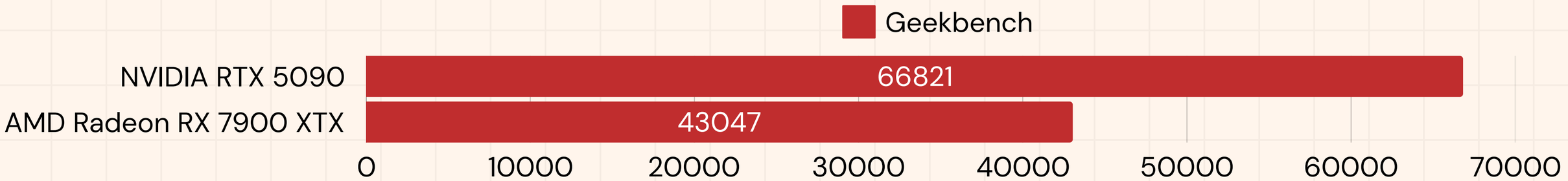



O PC QUE EU DISSE QUE COMPREI PRA FAZER OS TRABALHOS DO WESLEY
MINHA MÃE
EU

A GUERRA INTERMINÁVEL
NVIDIA VS AMD

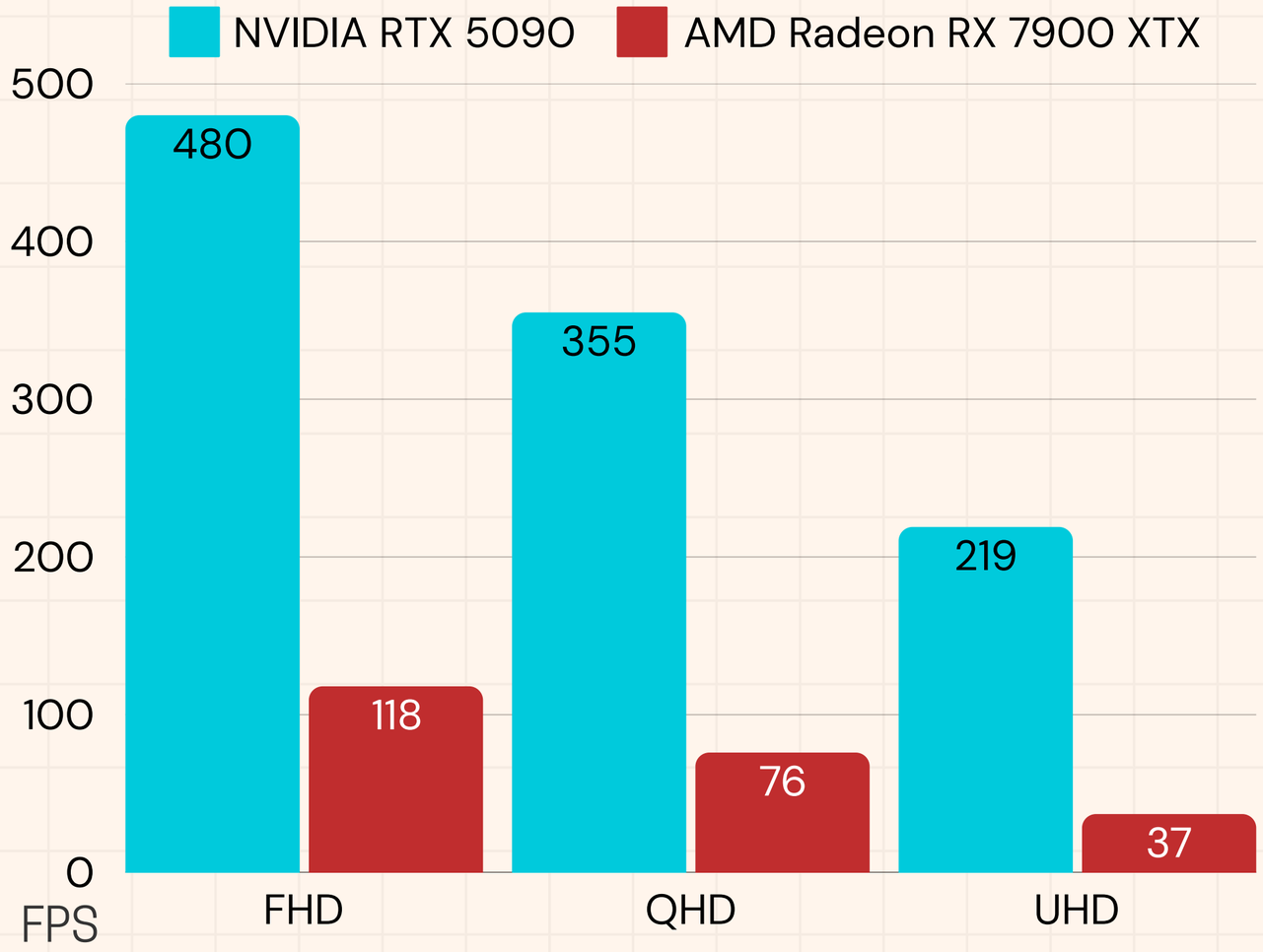
NVIDIA VS AMD





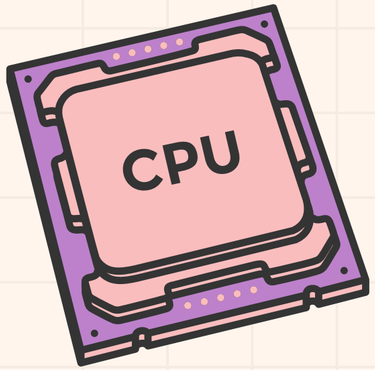
NVIDIA VS AMD

CYBERPUNK 2077



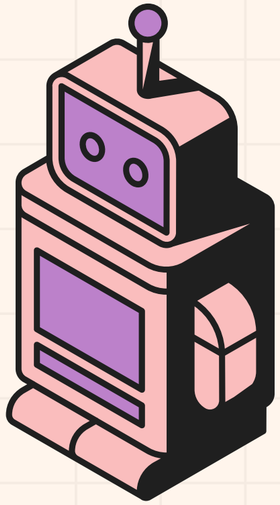
*TESTES FEITOS COM UM AMD RYZEN 9 7950X E 64GB DE RAM

TIPOS DE BENCHMARK



BENCHMARKS DE DESEMPENHO

Medem eficiência de sistemas, avaliando velocidade, processamento, recursos e escalabilidade. Usados para comparar soluções e identificar gargalos.



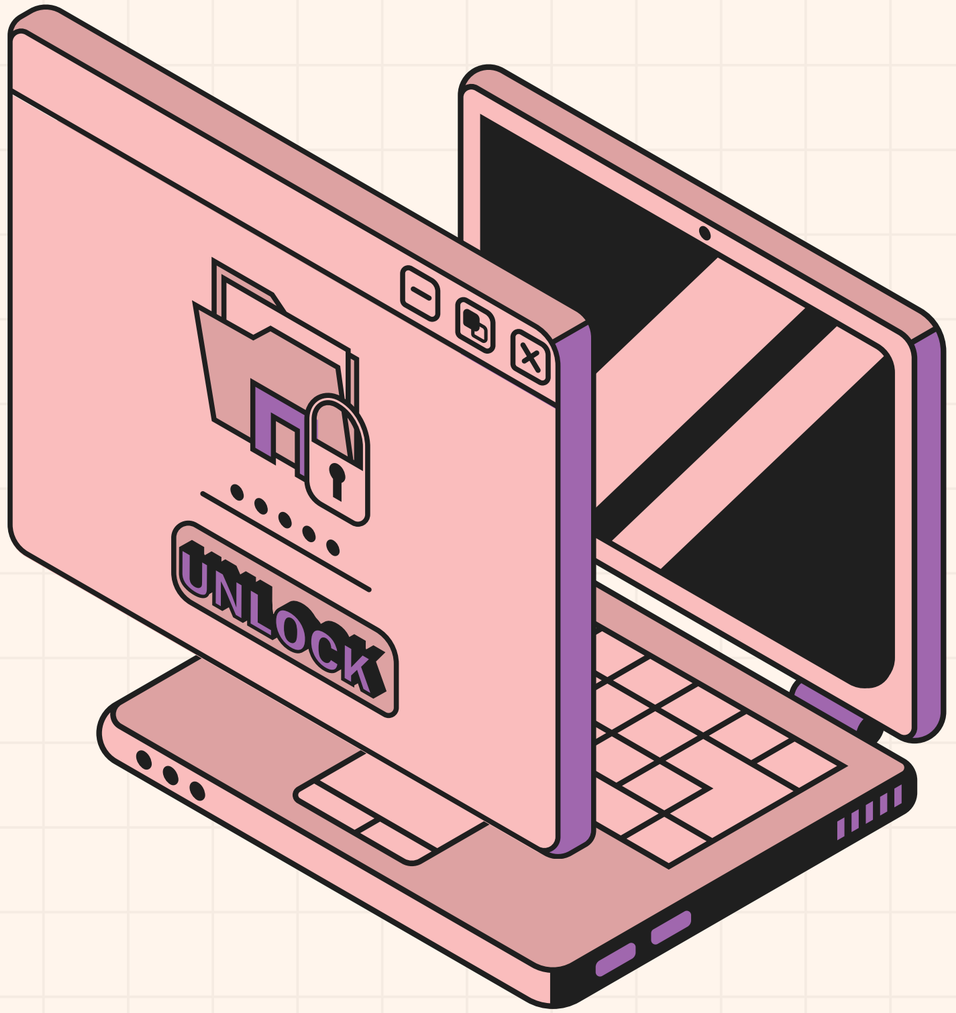
BENCHMARKS DE CORRETUDE

Avaliam precisão e consistência de sistemas, garantindo qualidade e confiabilidade.

BENCHMARK EM COMPILADORES

DESEMPENHO

Refere-se à eficiência com que um compilador realiza suas tarefas, tanto em termos de **tempo de compilação** quanto em termos da **qualidade do código gerado**.



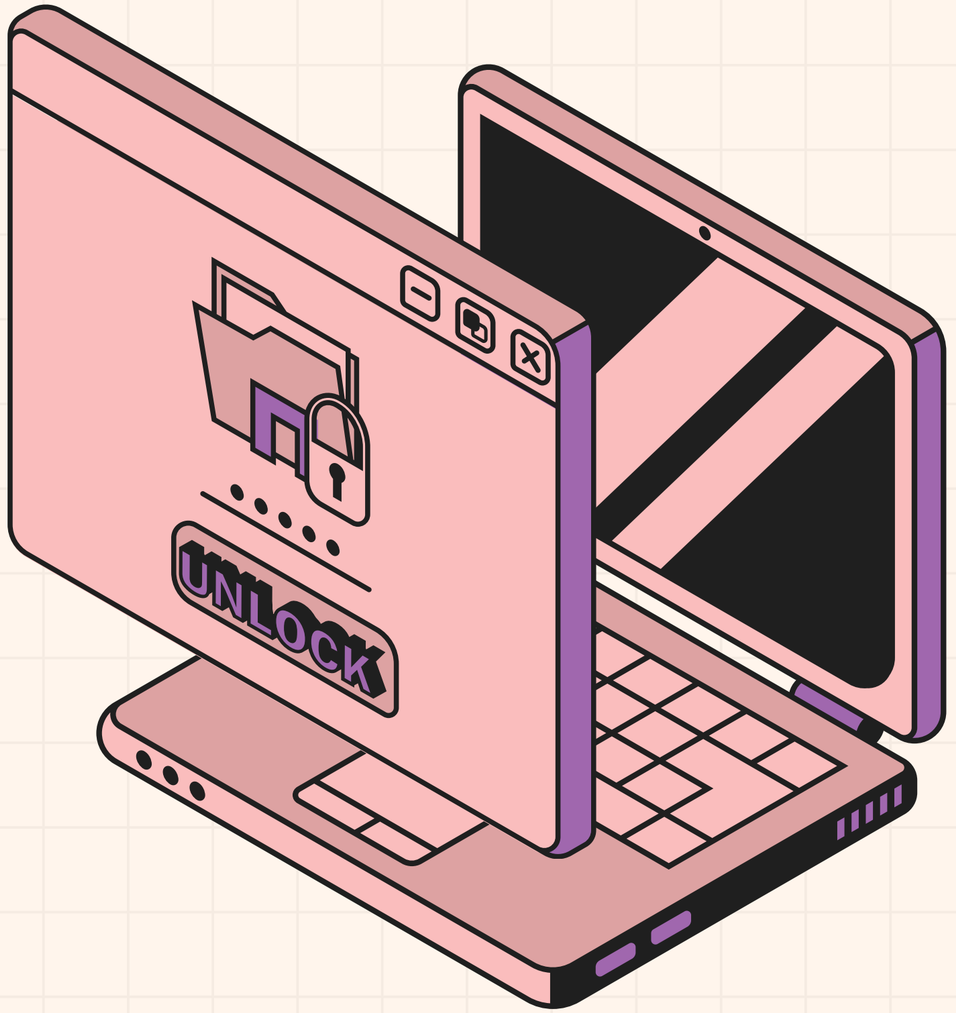
BENCHMARK EM COMPILADORES

DESEMPENHO

Tempo de compilação: Refere-se ao tempo que um compilador leva para transformar o código-fonte em código executável.

Fatores que alteram o tempo de compilação:

- Complexidade do código-fonte
- Fases do Compilador
- Arquitetura do Compilador
- Hardware e Ambiente



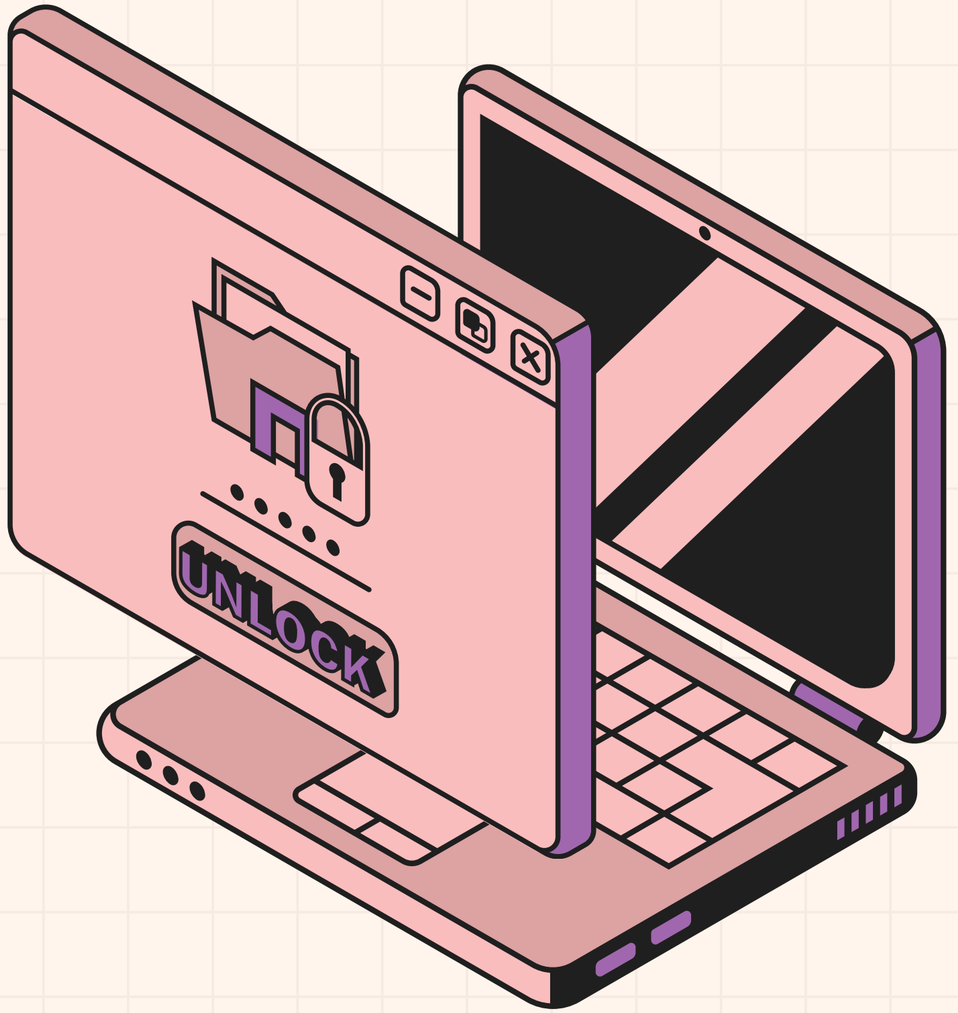
BENCHMARK EM COMPILADORES

DESEMPENHO

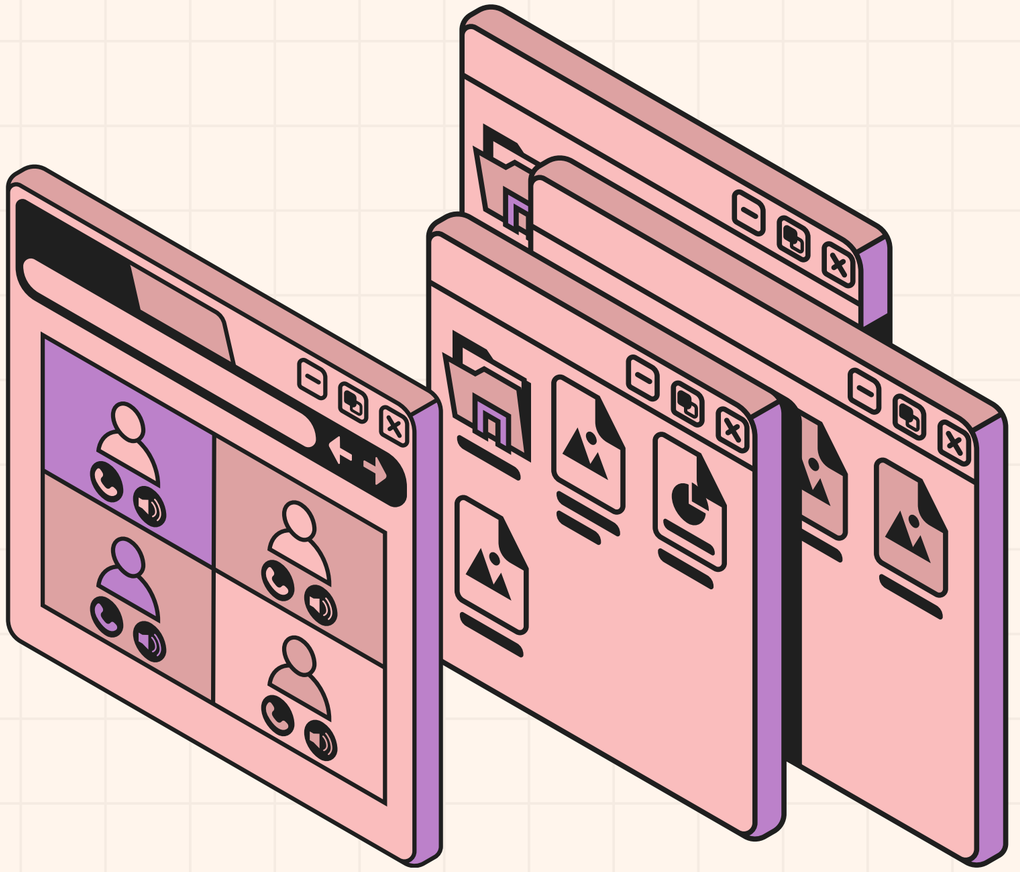
Eficiência do código gerado: Refere-se à qualidade do código de máquina produzido pelo compilador. Um código eficiente é aquele que executa rapidamente, consome poucos recursos e é compacto.

Métricas de eficiência:

- Tempo de execução
- Uso de memória
- Tamanho do código gerado
- Consumo e energia



BENCHMARK EM COMPILADORES



CORRETUDE

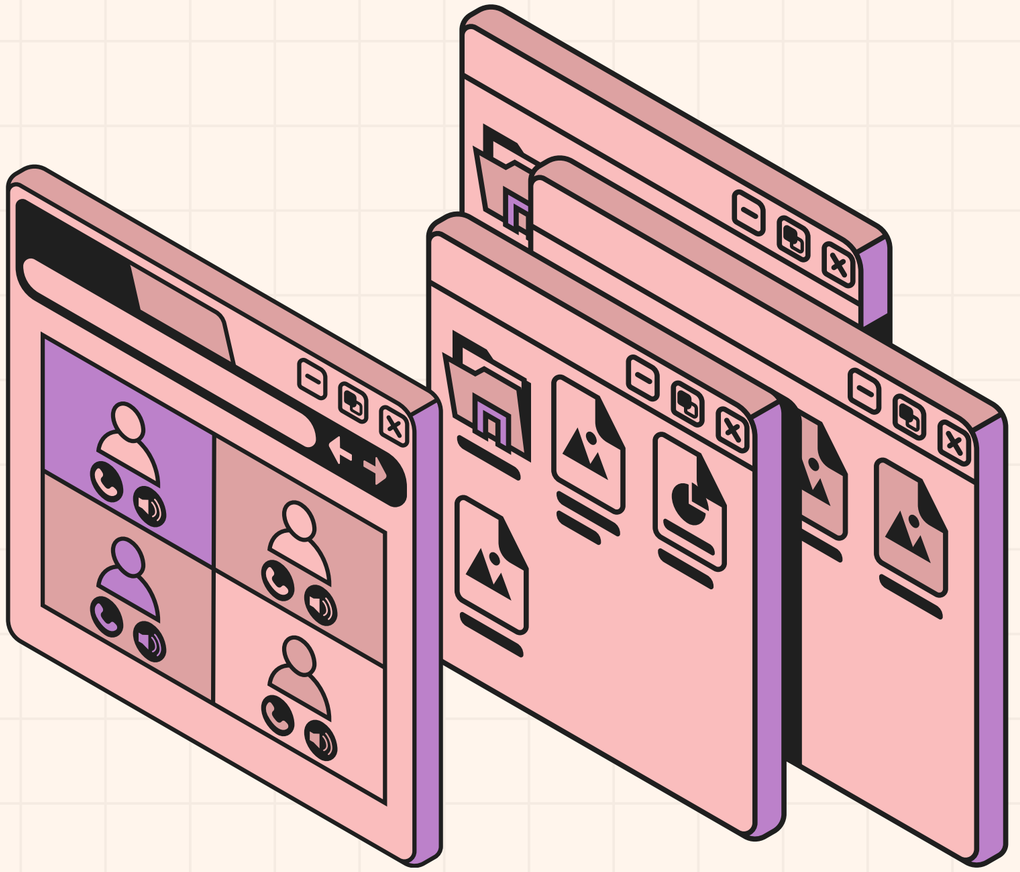
Refere-se à garantia de que o código gerado realiza exatamente o que o programador escreveu no código-fonte.

- Preservar a semântica
- Ausência de erros

Alguns desafios:

- Complexidade das Transformações e Otimizações
- Erros Sutis que Podem Passar Despercebidos

BENCHMARK EM COMPILADORES



CORRETUDE

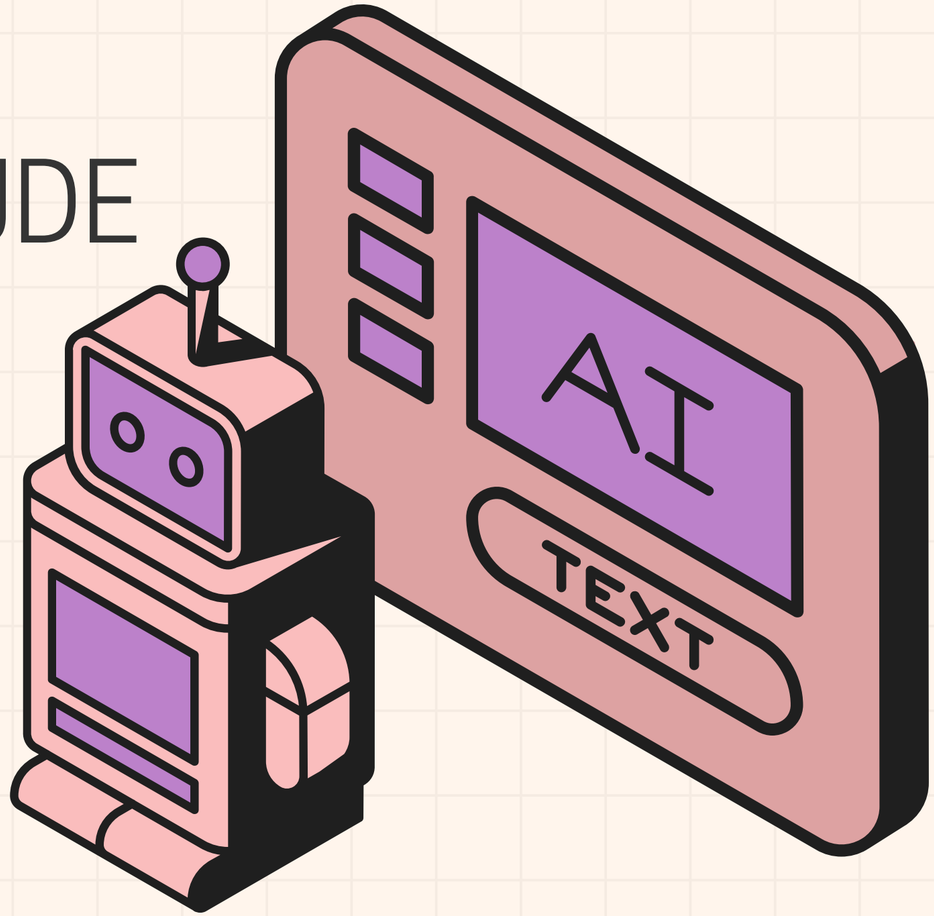
Como garantir?

- Verificação Formal de Compiladores
- Testes Exaustivos com Casos de Uso Variados
- Uso de Ferramentas de Análise Estática

Exemplos:

- CompCert
- CSmith

TRADE-OFF
DESEMPENHO & CORRETUDE

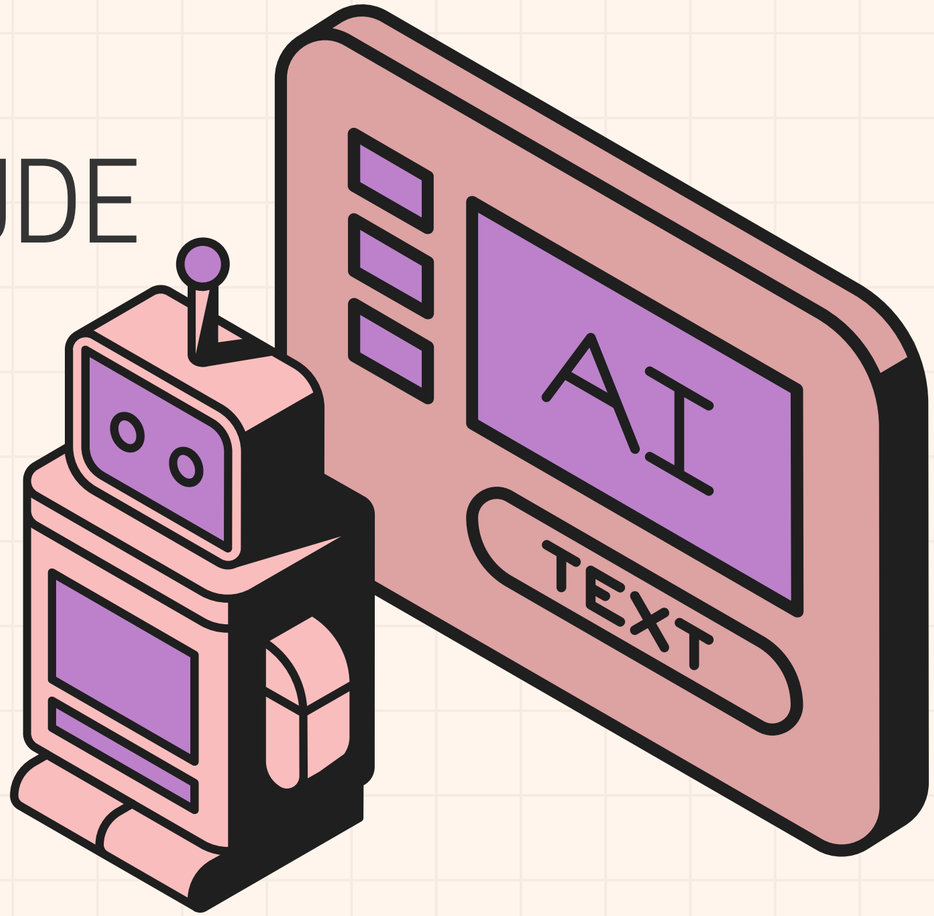


É um dos desafios mais importantes no desenvolvimento de compiladores. Esse equilíbrio é crucial porque, embora ambos os aspectos sejam essenciais, otimizar um pode impactar negativamente o outro.

O que compromete?

- Otimizações agressivas
- Verificações rigorosas

TRADE-OFF
DESEMPENHO & CORRETUDE



Como manter o equilíbrio?

- Uso de Técnicas de Verificação Durante o Desenvolvimento
- Testes de Desempenho e Corretude em Conjunto
- Otimizações Controladas e Graduais

MÉTRICAS & INDICADORES

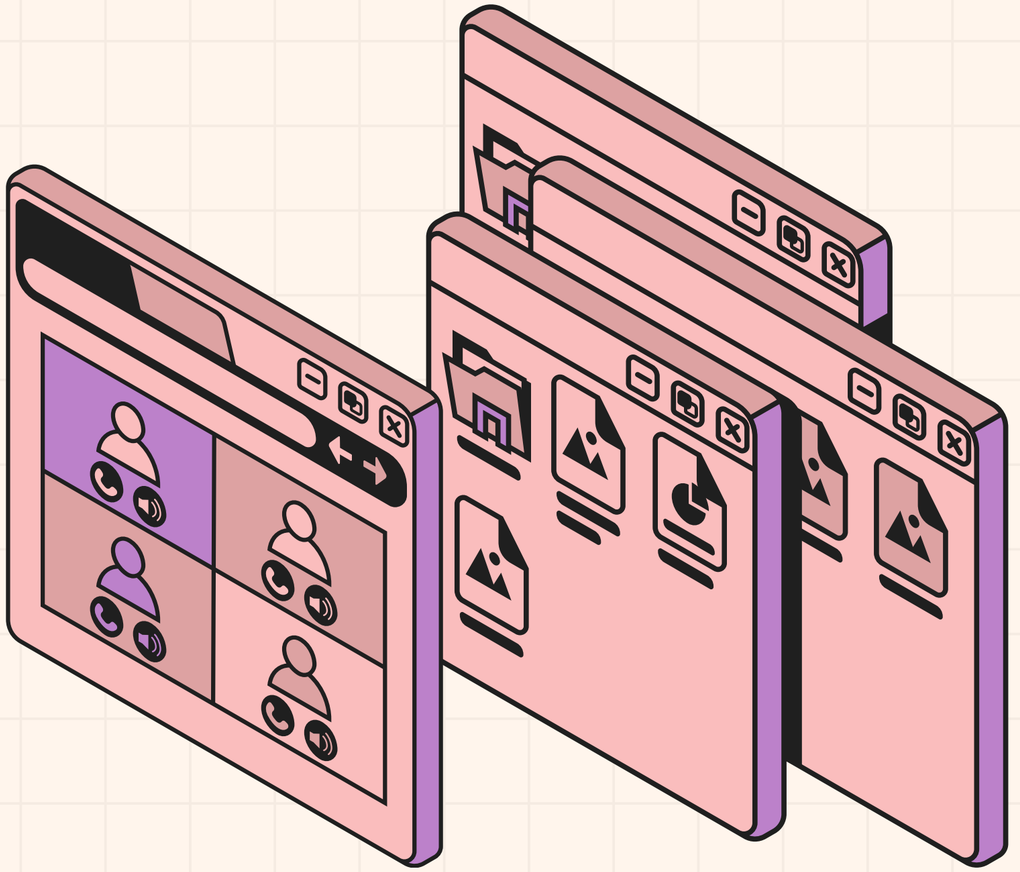
DESEMPENHO

Tempo de execução: Quanto tempo leva para concluir uma tarefa

Throughput: Número de operações concluídas por unidade de tempo

Latência: Tempo entre uma solicitação e uma resposta

Uso de recursos: CPU, memória, disco, rede



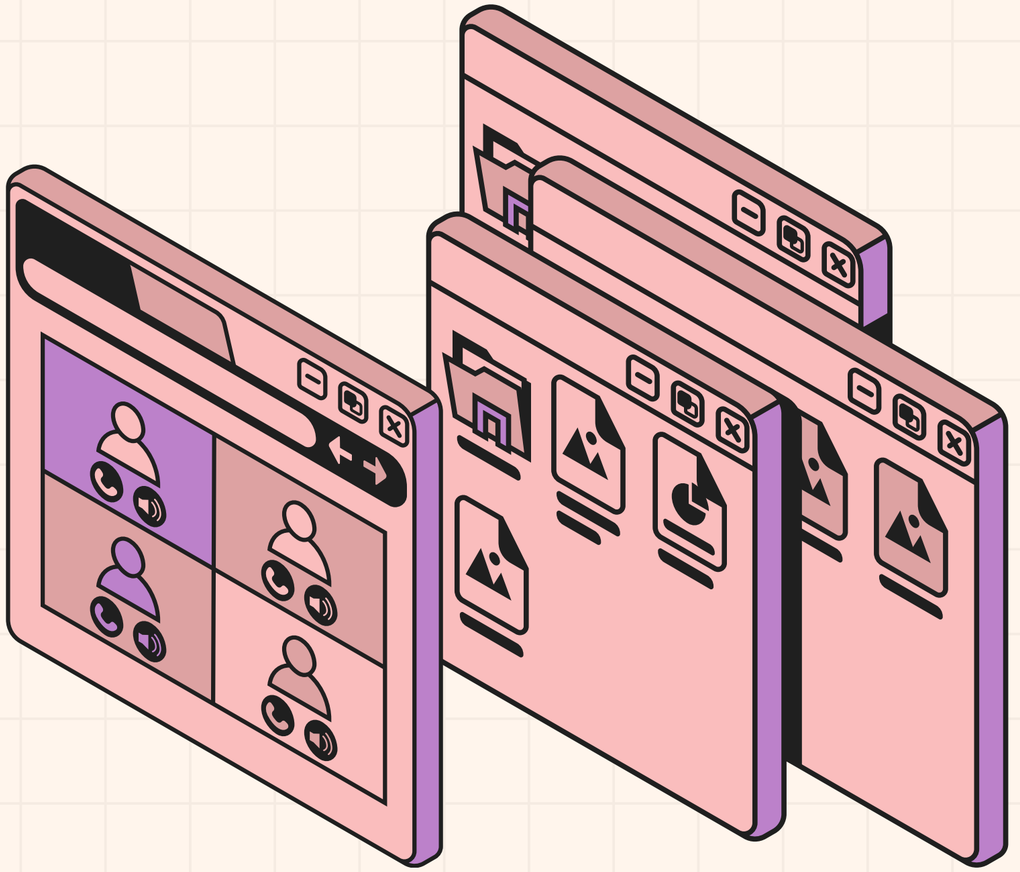
MÉTRICAS & INDICADORES

CORRETUDE

Taxa de Erros Detectados: quantidade de erros identificados durante a compilação ou execução do código gerado.

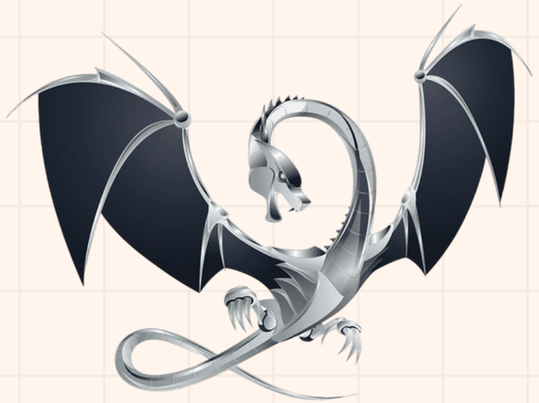
Taxa de Sucesso em Testes: porcentagem de testes que passam sem erros.

Número de Bugs Reportados: quantidade de bugs relatados pelos usuários do compilador.



GCC VS CLANG
COMPARAÇÃO

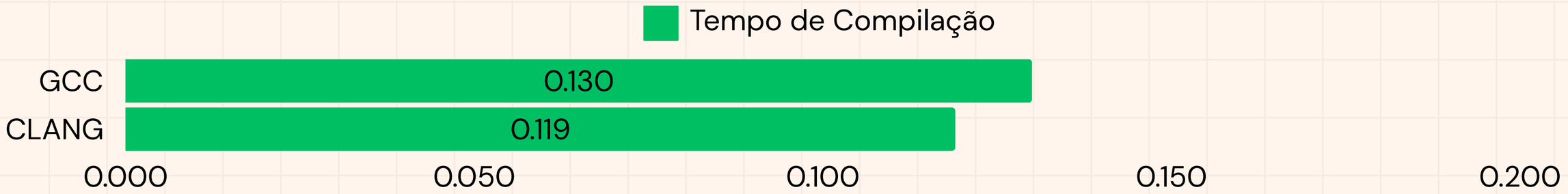
- Código otimizado
- Alta compatibilidade
- Maior velocidade de execução
- Estabilidade

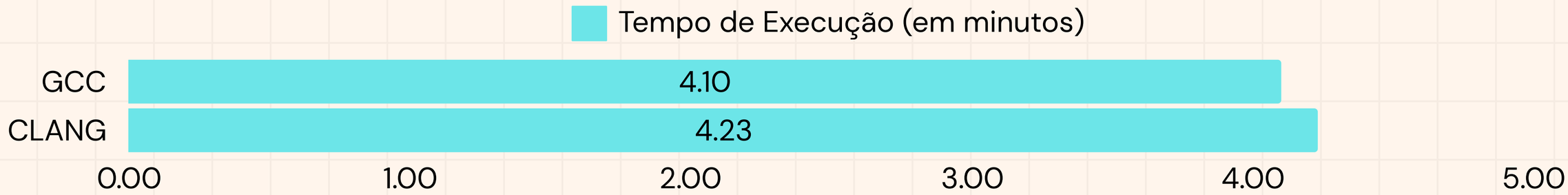


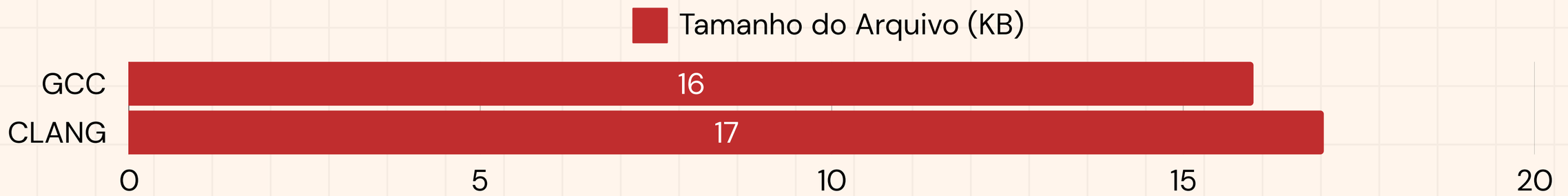
- Mensagens de erro claras e diagnósticos avançados
- Integração com ferramentas modernas
- Maior velocidade de compilação

GCC VS CLANG

TESTES EM CÓDIGO C COM CONTAS MATEMÁTICAS

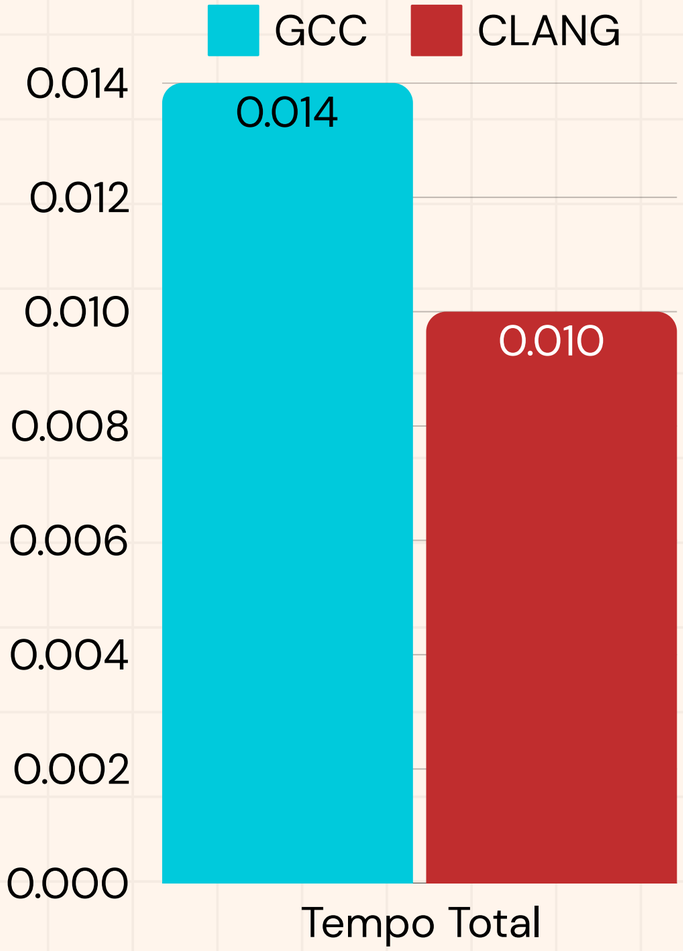


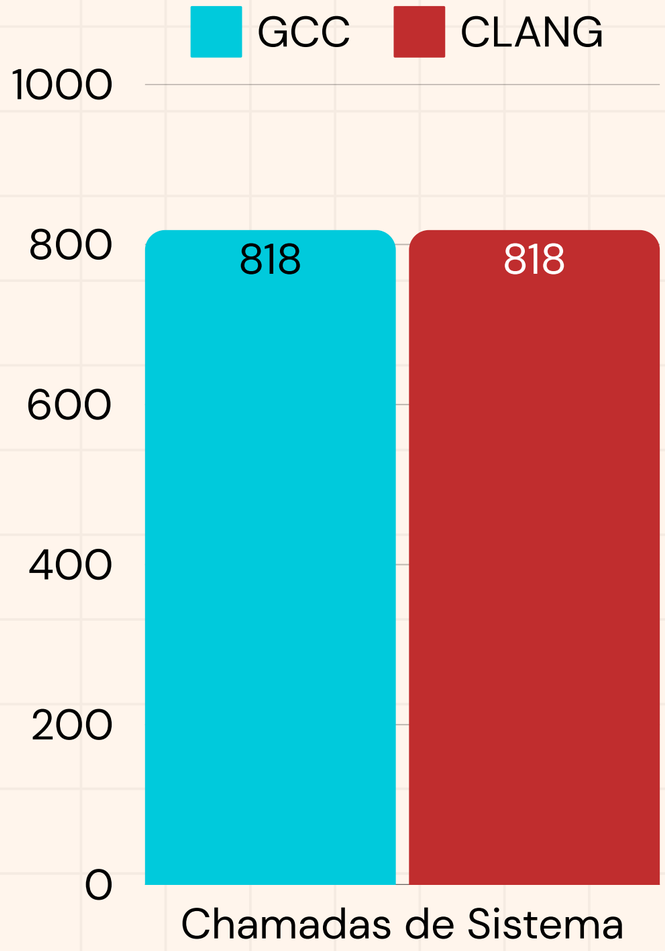




GCC VS CLANG

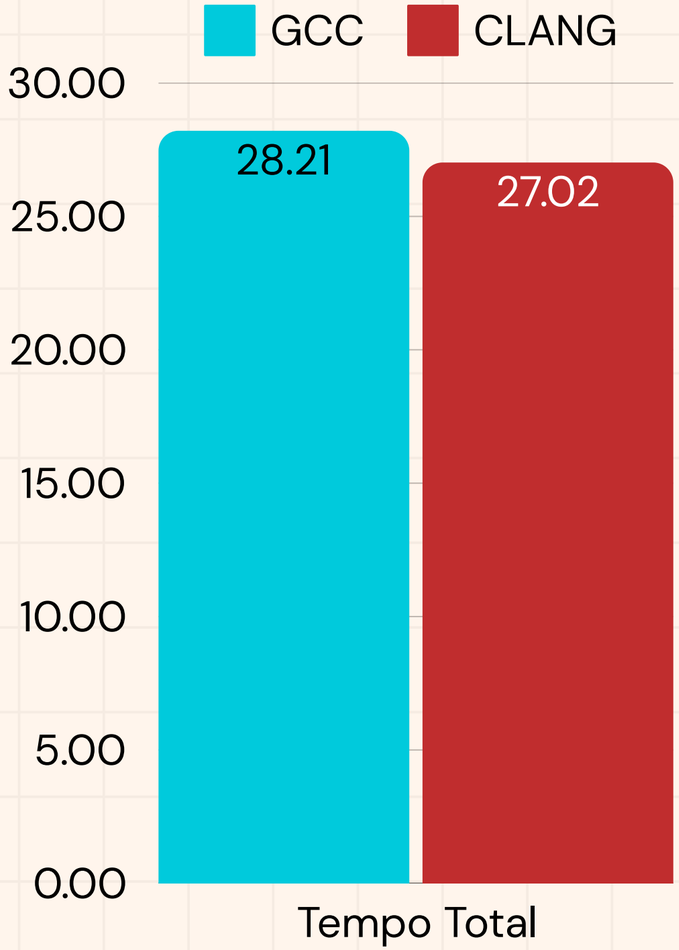
STRACE PARA CHAMADAS DE SISTEMA

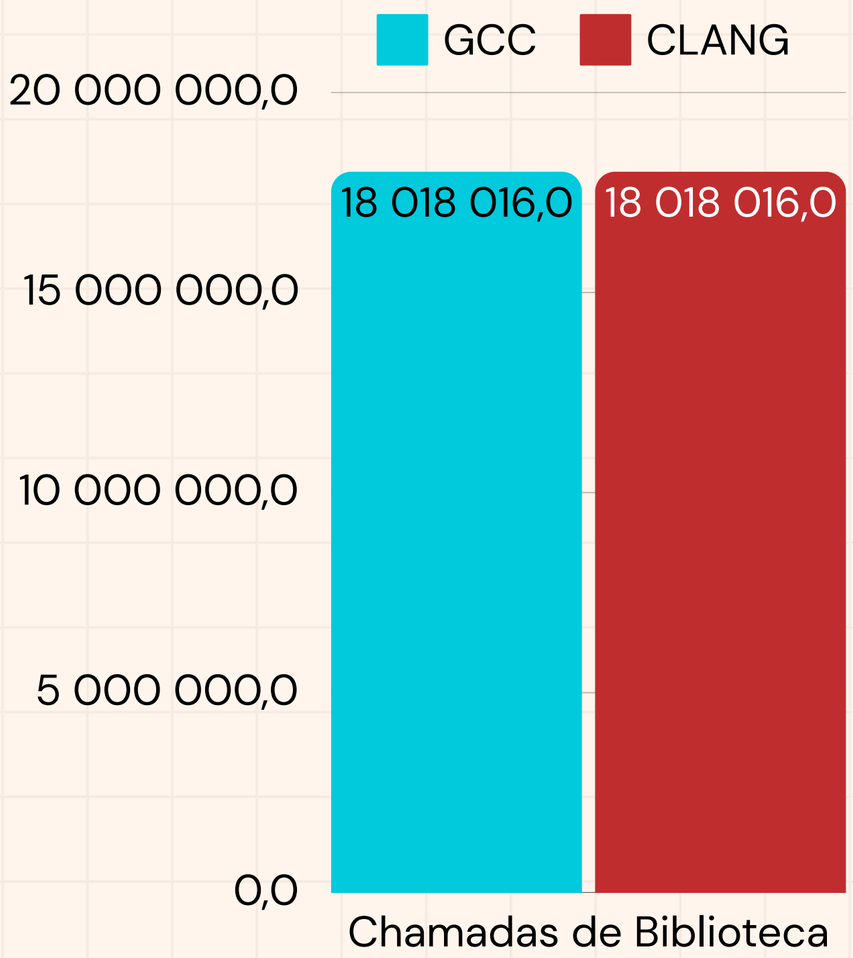




GCC VS CLANG

LTRACE PARA CHAMADAS DE BIBLIOTECA

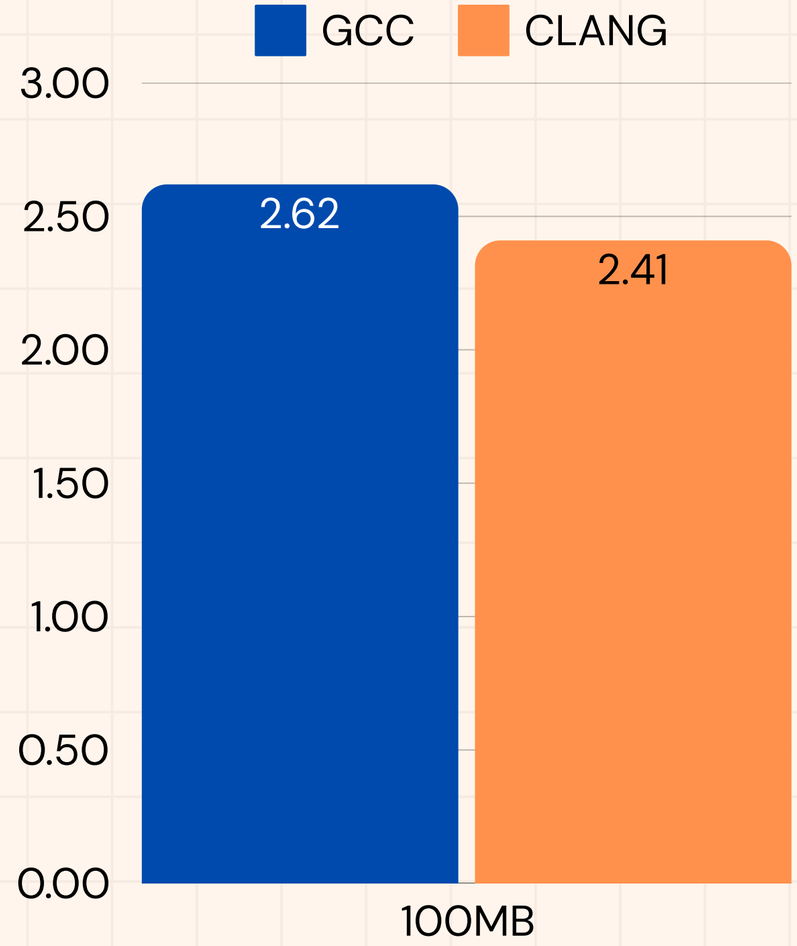


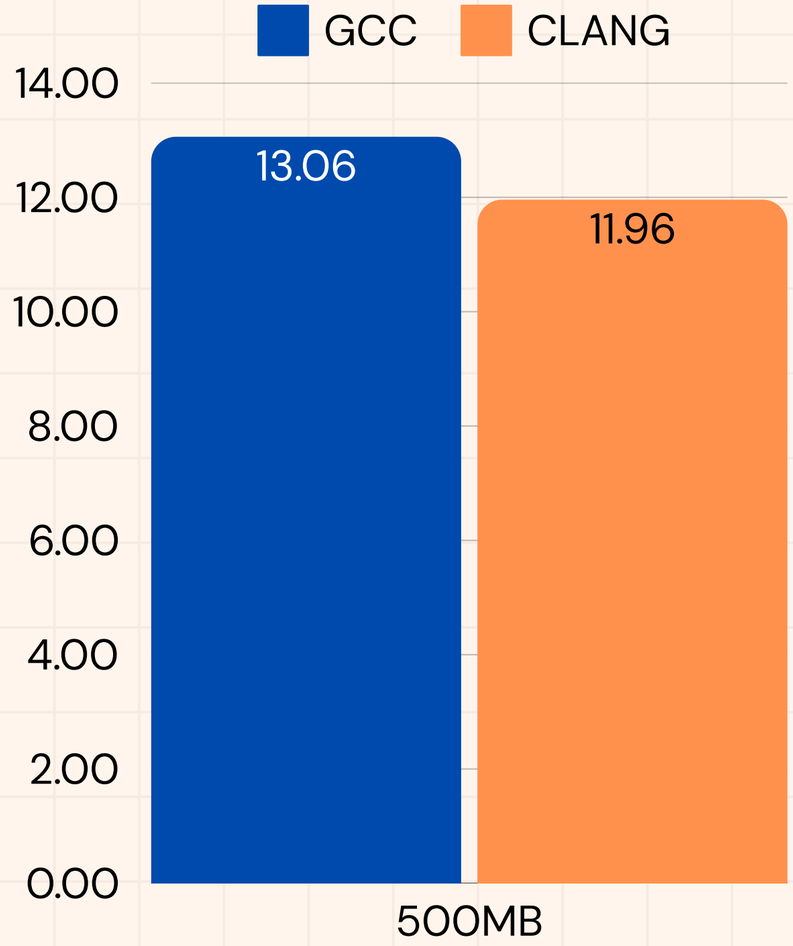


*TEMPO EM MINUTOS

GCC VS CLANG

GZIP

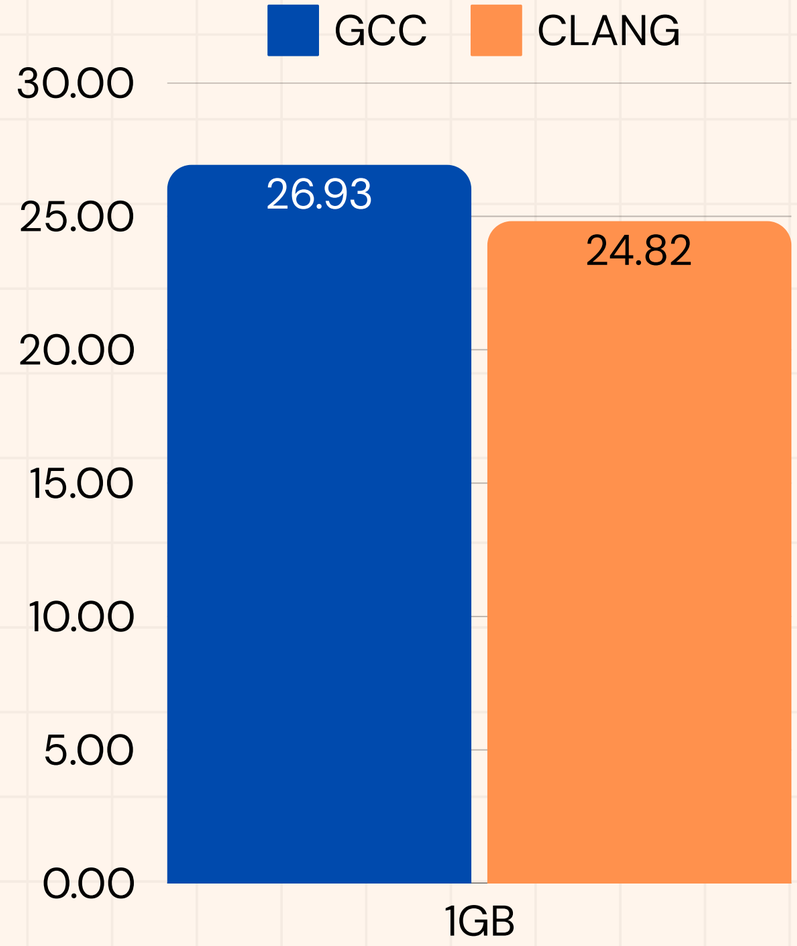


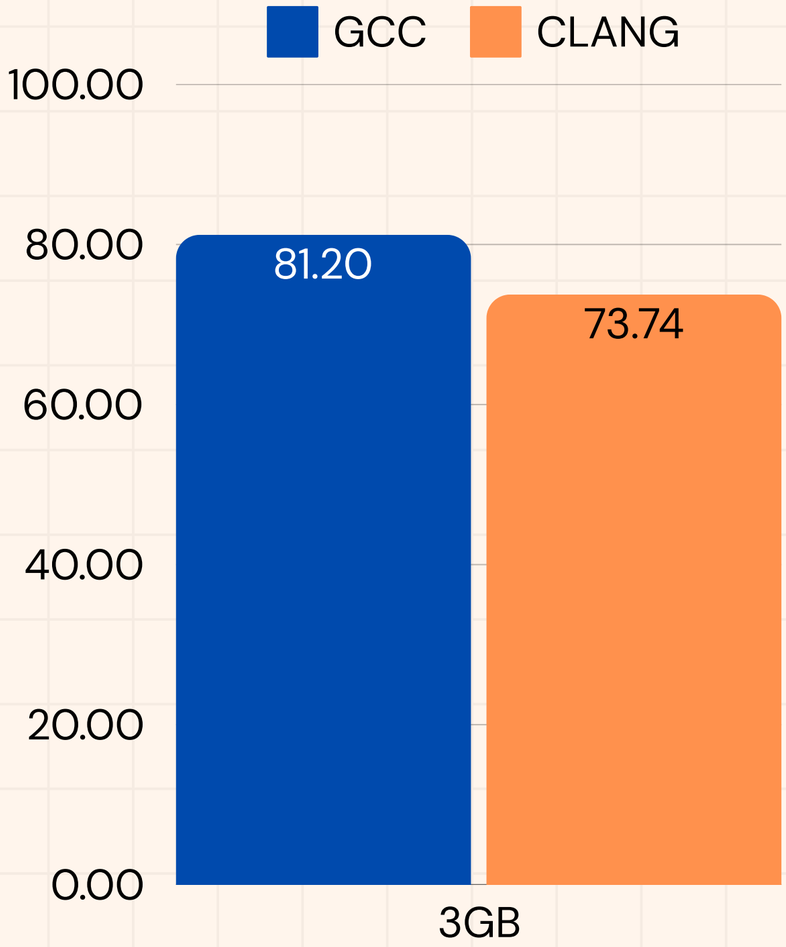


*TEMPO EM SEGUNDOS

GCC VS CLANG

GZIP





*TEMPO EM SEGUNDOS

FUTURO DOS BENCHMARKS

COMPUTAÇÃO QUÂNTICA

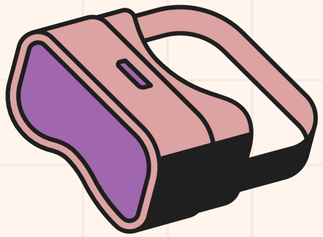
QED e Quantum Volume são usados para avaliar qubits e portas quânticas.

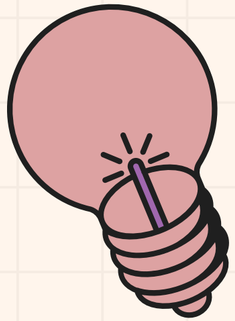
INTELIGÊNCIA ARTIFICIAL

MLPerf e DAWNBench são usados para comparar desempenho em treinamento e inferência de modelos de IA.

INTERNET DAS COISAS (IOT)

foco em baixo consumo de energia e latência para dispositivos IoT.





OBRIGADO

PELA ATENÇÃO!

CONTAMOS COM SEUS VOTOS!