

Monografia: Cache Wizard

CacheWizard

Trabalho 2 da disciplina Organização de Computadores (2023.1)

O projeto "CacheWizard" é uma ferramenta de automação desenvolvida em Python para simplificar e otimizar o processo de endereçamento da cache em sistemas computacionais.

Desenvolvedores

Nome	Numero USP
Victor Cologni Seles	11795398
Eric Rodrigues das Chagas	12623971
Ana Beatriz Araujo Ferreira	12678044
Gabriel Faccini Gregoris	13672936

Descrição

O projeto "CacheWizard" é uma ferramenta de automação desenvolvida em Python para simplificar e otimizar o processo de endereçamento da cache em sistemas computacionais.

Através de uma interface intuitiva e amigável, o CacheWizard permite aos usuários definir parâmetros e configurações relacionadas à cache, como tamanho, associatividade e tamanho do bloco.

Com base nessas informações, o CacheWizard gera automaticamente o mapeamento de endereços para a cache, simplificando assim o trabalho manual e propenso a erros.

O CacheWizard tem como objetivo simplificar e agilizar o processo de configuração da cache, permitindo que os usuários se concentrem em tarefas mais avançadas e complexas. Com sua abordagem automatizada e recursos adicionais de aprendizado, espera-se que o projeto ofereça uma solução eficiente e de fácil utilização para aprimorar o entendimento e a implementação do endereçamento da cache em sistemas computacionais.

Como usar

Abrindo o programa

Você pode rodar o programa de duas formas

⇒ Você poder usar o comando do linux pelo terminal (estando na pasta root do programa)

```
python3 '__main__ .py'
```

Exemplo:

```
MINGW64;c:/Users/ericr/Desktop/USP_temp/OrgComp/Trabalho 02/CacheWizard

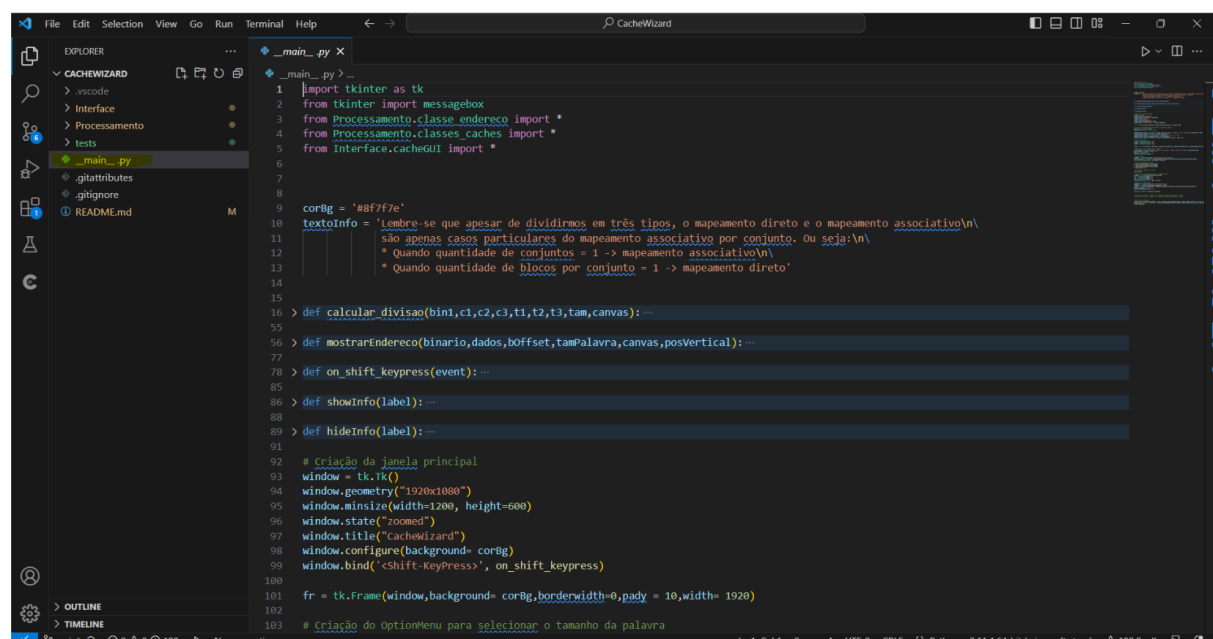
ericr@Eric_IdeapadGam MINGW64 ~/Desktop/USP_temp/OrgComp/Trabalho 02/CacheWizard (main)
$ ls
Interface/  Processamento/  README.md  '__main__ .py'  tests/

ericr@Eric_IdeapadGam MINGW64 ~/Desktop/USP_temp/OrgComp/Trabalho 02/CacheWizard (main)
$ python3 '__main__ .py'|
```

⇒ Abrir o seu editor de texto e rodar dentro dele mesmo

Segue um exemplo abaixo usando o vscode

Abrindo o arquivo `__main__ .py` basta apertar **F5** que o programa rodará



Usando o programa

Ao abrir o programa, a seguinte tela aparecerá:

Tamanho da Palavra: 16

Memória Cache L1: Palavras por Bloco: 2 Blocos por Conjunto: 1 Conjuntos: 4

Memória Cache L2: Palavras por Bloco: 2 Blocos por Conjunto: 2 Conjuntos: 2

Memória Cache L3: Palavras por Bloco: 4 Blocos por Conjunto: 16 Conjuntos: 1

Endereço (em decimal): 341

Calcular

0000001110110100

0000001110110100

0000001110110100

cache L1	cache L2	cache L3
Byte Offset: 1 bits	Byte Offset: 1 bits	Byte Offset: 1 bits
Word Offset: 1 bits	Word Offset: 1 bits	Word Offset: 2 bits
Index: 2 bits	Index: 3 bits	Index: 0 bits
Tag: 12 bits	Tag: 11 bits	Tag: 13 bits

Num geral, o programa é bastante intuitivo

Precisamos apenas colocar os valores em cada uma das caches que ele retornara o resultado

Uma observação importante é que definimos a diferença entre, Mapeamento Livrementemente Associativo, Associativo por conjunto e Mapeamento direto de acordo com o valor dos conjuntos de cada cahce. Desta forma segue abaixo o conceito utilizado:

- Quantidade de conjuntos = 1: Livrementemente associativo
- Quantidade de conjuntos > 1: Associativa por conjunto
- Quantidade de blocos por conjunto = 1: Mapeamento direto