

Aluno: Alexandre Junior Lelis Rodrigues

Programa escrito
em Golang
Código

```
-----  
package main  
  
import "fmt"  
  
func main() {  
    // Solicita o número de quadros ao usuário  
    var quadros int  
    fmt.Print("Digite o número de quadros: ")  
    fmt.Scanln(&quadros)  
  
    // Solicita a sequência de números ao usuário  
    fmt.Print("Digite a sequência de números separados por espaço: ")  
    Sequence := lerSequencia()  
  
    // Chama a função para acessar a memória com os quadros e a  
    sequência fornecidos  
    acessarMemoria(quadros, Sequence)  
}  
  
// Função para ler a sequência de números fornecida pelo usuário  
func lerSequencia() []int {  
    var elemento int  
    var sequencia []int  
    for {  
        _, err := fmt.Scanf("%d", &elemento) // Lê um elemento  
        if err != nil { // Se ocorrer um erro (como EOF), encerra o loop  
            break  
        }  
        sequencia = append(sequencia, elemento) // Adiciona o elemento  
à sequência  
    }  
    return sequencia // Retorna a sequência lida  
}
```

```

// Função para acessar a memória com os quadros e a sequência fornecidos
func acessarMemoria(quadros int, Sequence []int) {

    current := []int{} // Inicializa a lista que representa os quadros atuais
    toLeave := 0       // Índice para o quadro a ser substituído
    pageFaults := 0    // Contador de page faults

    // Preenche a lista `current` com -1 para indicar que os quadros estão
vazios
    for i := range current {
        current[i] = -1
    }

    // Itera sobre a sequência de acessos à memória
    for _, value := range Sequence {
        if !contains(current, value) { // Se o valor não estiver nos quadros
atuais
            if len(current) < quadros { // Se ainda houver espaço nos
quadros
                current = append(current, value) // Adiciona o valor
ao final dos quadros
            } else { // Se todos os quadros estiverem ocupados
                current[toLeave] = value // Substitui o quadro
indicado por `toLeave` pelo novo valor
                toLeave++                // Atualiza o índice para o
próximo quadro a ser substituído
            }
            if toLeave == quadros { // Se `toLeave` alcançar o
número de quadros, volta ao primeiro quadro
                toLeave = 0
            }
        }
        pageFaults++                // Incrementa o contador
de page faults
        fmt.Print(current)          // Imprime os quadros
atuais
        fmt.Print("| Page Fault")   // Indica que ocorreu um
page fault
    } else {
        fmt.Print(current) // Imprime os quadros atuais
    }
}

```

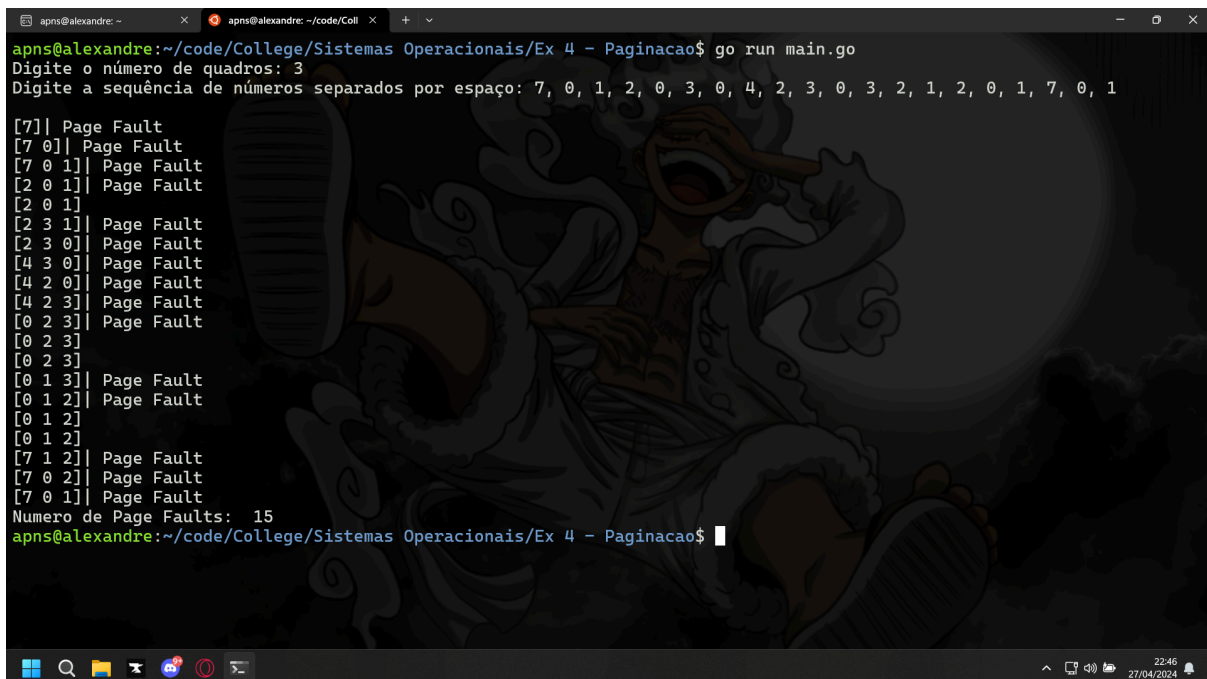
```

        fmt.Println() // Pula para a próxima linha
    }

    // Imprime o número total de page faults
    fmt.Println("Numero de Page Faults: ", pageFaults)
}

// Função para verificar se um valor está presente em uma lista de inteiros
func contains(current []int, value int) bool {
    for _, currentValue := range current {
        if currentValue == value {
            return true
        }
    }
    return false
}

```



```

apns@alexandre: ~/code/College/Sistemas Operacionais/Ex 4 - Paginacao$ go run main.go
Digite o número de quadros: 3
Digite a sequência de números separados por espaço: 7, 0, 1, 2, 0, 3, 0, 4, 2, 3, 0, 3, 2, 1, 2, 0, 1, 7, 0, 1

[7] Page Fault
[7 0] Page Fault
[7 0 1] Page Fault
[2 0 1] Page Fault
[2 0 1]
[2 3 1] Page Fault
[2 3 0] Page Fault
[4 3 0] Page Fault
[4 2 0] Page Fault
[4 2 3] Page Fault
[0 2 3] Page Fault
[0 2 3]
[0 2 3]
[0 1 3] Page Fault
[0 1 2] Page Fault
[0 1 2]
[0 1 2]
[7 1 2] Page Fault
[7 0 2] Page Fault
[7 0 1] Page Fault
Numero de Page Faults: 15
apns@alexandre:~/code/College/Sistemas Operacionais/Ex 4 - Paginacao$

```

O código está disponível também no github

<https://github.com/a3ylf/College/tree/master/Sistemas%20Operacionais/Ex%2004%20-%20Paginacao>

