


 PCS 2302/2024 Laboratório de Fundamentos da Eng.de Computação	PCS-2302 / PCS-2024 Lab. de Fundamentos de Eng. de Computação
	Aula 05 Exercícios Turma 3 Professores: Marcos A. Simpício Junior Paulo Sergio Muniz Silva

Aula 04:
Introdução ao
Paradigma de
Objetos
Autores:
Anna H. R. Costa
Jaime S. Sichman
João José Neto
Paulo S. Muniz Silva
Ricardo L. A. Rocha
Reestruturação:
Paulo S. Muniz Silva
v.1.0 ago. 2012

   PCS 2302/2024 Laboratório de Fundamentos da Eng.de Computação	Exercícios - Objetivos
	<ul style="list-style-type: none">• Praticar a interação entre o “hardware” da MVN (código em Java) e o software por ela executado (escrito em linguagem de montagem da MVN)• Introduzir o uso das instruções Get Data (GD), Put Data (PD) e Operating System (OS), não utilizadas até o momento

Aula 04:
Introdução ao
Paradigma de
Objetos
Autores:
Anna H. R. Costa
Jaime S. Sichman
João José Neto
Paulo S. Muniz Silva
Ricardo L. A. Rocha
Reestruturação:
Paulo S. Muniz Silva
v.1.0 ago. 2012

Operações de Entrada-Saída da MVN

Formato da instrução:

OP	Tipo	Dispositivo (LU)
4 bits	4 bits	8 bits

OP D (entrada: GD) ou E (saída: PD)

Tipo Tipos de dispositivo:

- 0 = Teclado
- 1 = Monitor
- 2 = Impressora
- 3 = Disco

Dispositivo Identificação do dispositivo. Pode-se ter vários tipos de dispositivo, ou unidades lógicas (LU). No caso do **disco**, um **arquivo** é considerado uma unidade lógica.

Pode-se ter, portanto, até 16 tipos de dispositivos e, cada um, pode ter até 256 unidades lógicas.

Ex: GD /300 ; Lê 16 bits do disco cuja unidade lógica é 00.

; Resultado é colocado no acumulador

PD /100 ; Escreve conteúdo do acumulador no monitor cuja
unidade lógica é 00.

Operações de Entrada-Saída da MVN

- Para adicionar/remover dispositivos manualmente na MVN: **comando "s"**
- Para carregar dispositivos automaticamente, pode ser usado o **arquivo "disp.lst"**

- Sintaxe e exemplos:

Tipo LU nome_classe [nome_arquivo tipo_acesso]

0 0 mvn.dispositivo.Teclado

1 0 mvn.dispositivo.Monitor



3 0 mvn.dispositivo.Disco arq.txt b

3 1 mvn.dispositivo.Disco arq.txt e

→ Disco 0: leitura/escrita

Disco 1: apenas escrita ←

- Obs.: coloque-o na raiz do projeto, se estiver usando o IDE (Eclipse/Netbeans) ou na mesma pasta que o .jar se estiver executando via prompt de comando

DEPARTAMENTO DE ENGENHARIA DE
COMPUTAÇÃO E SISTEMAS DIGITAIS



PCS 2302/2024
Laboratório de
Fundamentos da
Eng.de Computação

Aula 04:
Introdução ao
Paradigma de
Objetos

Autores:
Anna H. R. Costa
Jaime S. Sichman
João José Neto
Paulo S. Muniz Silva
Ricardo L. A. Rocha

Reestruturação:
Paulo S. Muniz Silva

v.1.0 ago. 2012

Chamada de Supervisor (SVC) na MVN

- Relembrando:

OS	Parâmetros	Operação
----	------------	----------

Exemplo: OS /01FF



OS Instrução F (1 byte)

Parâmetros Número de parâmetros (1 byte)


Operação Código de operação [00 a FF] (2 bytes)
- Parâmetros devem ser posicionados no endereço de memória anterior ao da chamada, na forma de pilha (primeiro parâmetro vem logo acima da chamada OS)

Ex.: operação FF com dois parâmetros.

Passando ULs 1 e 2 como parâmetros:	UL2	K	/0002
	UL1	K	/0001
		OS	/02FF

DEPARTAMENTO DE ENGENHARIA DE
COMPUTAÇÃO E SISTEMAS DIGITAIS



PCS 2302/2024
Laboratório de
Fundamentos da
Eng.de Computação

Aula 04:
Introdução ao
Paradigma de
Objetos

Autores:
Anna H. R. Costa
Jaime S. Sichman
João José Neto
Paulo S. Muniz Silva
Ricardo L. A. Rocha

Reestruturação:
Paulo S. Muniz Silva

v.1.0 ago. 2012

Exercício 01 (MVN)

TYGXXA05E01_rotinas.asm (Obrigatório)

Primitiva da biblioteca elementar da MVN a ser desenvolvida na “linguagem de máquina” do simulador MVN: **GETLINEF**

Função: ler uma linha de um arquivo-texto, visto como um dispositivo do tipo “Disco” pela MVN

GETLINEF lê apenas uma linha de texto, ou seja, a leitura termina quando é encontrada a palavra que indica final da linha (EOL) ou final de arquivo (EOF)

Pergunta: qual palavra deve ser usada como EOL? E como EOF?

Os caracteres contidos nas palavras (2 bytes) da linha são escritos num *buffer* na memória –uma *string*. O **EOL** e **EOF** **não devem ser escritos** no buffer. O final do string deve ser indicado usando uma palavra de finalização (**EOS: /0000**), que **deve sempre ser escrita** no buffer.



Exercício 01 (MVN)

TYGXXA05E01_rotinas.asm (Obrigatório)

GETLINEF (continuação).

O buffer tem um tamanho máximo de palavras. Linhas de tamanho maior do que o especificado devem ser truncadas. (ex.: para reservar 16 palavras, use o comando "\$ /0010")

Retorna 1 (*true*) se não chegar ao final do arquivo (EOF); retorna 0 (*false*) se chegar ao final do arquivo.

Nesta primeira versão de GETLINEF – de escopo limitado – as linhas do arquivo-texto têm número par de caracteres.

Parâmetros: o endereço do *buffer* (GL_END), a unidade lógica do arquivo-texto (GL_UL) e o tamanho do buffer (GL_BUF).

Retorno (acumulador): 0 (*false*) se chegar ao final do arquivo e 1 (*true*) no caso contrário.



Exercício 01: Entrega

TYGXXA05E01_rotinas.asm (Entrega obrigatória)

- Contém o GETLINEF e, possivelmente, todas as rotinas desenvolvidas nas aulas anteriores

TYGXXA05E01_const.asm (Entrega obrigatória)

- Contém todas as constantes usadas tanto pelo programa principal como pelas rotinas

TYGXXA05E01_main.asm (Entrega obrigatória)

- Use o arquivo TYGXXA05E01_main.asm fornecido:




Endereço de início do programa principal: 0000




Obs.: basta colocar o main como primeiro arquivo a ser ligado e fixar a base de relocação em 0 para obter esse endereço.



Endereço com unidade lógica (GL_UL): 0002

Endereço com tamanho do buffer (GL_BUF): 0004

Endereço do buffer (GL_END): 0006

  <p>DEPARTAMENTO DE ENGENHARIA DE COMPUTAÇÃO E SISTEMAS DIGITAIS</p>  <p>PCS 2302/2024 Laboratório de Fundamentos da Eng.de Computação</p> <p>Aula 04: Introdução ao Paradigma de Objetos</p> <p>Autores: Anna H. R. Costa Jaime S. Sichman João José Neto Paulo S. Muniz Silva Ricardo L. A. Rocha</p> <p>Reestruturação: Paulo S. Muniz Silva</p> <p>v.1.0 ago. 2012</p>	<h2>Exercício 02 :</h2> <h3>Chamada de Supervisor (SVC)</h3>
	<p>TYGXXA05E02 (Obrigatório)</p> <p>Chamada SVC (em Java).</p> <p>Altere o código da MVN para que a partir de uma <i>Chamada de Supervisor</i> (instrução OS) seja executada a função reset. Essa função posiciona o cursor de leitura da unidade lógica de disco em seu início.</p> <p>Parâmetro: lu – o número da unidade lógica de I/O</p> <p>Retorno: 0 em caso de sucesso, -1 em caso de erro (número da unidade lógica inválida, inexistente ou somente de escrita)</p> <p>Instrução de Chamada: OS /01FF</p> <p>OBS: O valor do parâmetro a ser passado para a chamada de supervisor deve ser posicionado no endereço de memória anterior ao da chamada.</p> <p>Restrições: Funciona somente se a unidade lógica estiver aberta para leitura e for uma unidade de disco.</p> <p>IMPORTANTE: Não existe uma operação explícita de reset de arquivo no Java, portanto é necessário pensar em alguma forma para executá-la (ex.: fechar e abrir o arquivo)</p>

  <p>DEPARTAMENTO DE ENGENHARIA DE COMPUTAÇÃO E SISTEMAS DIGITAIS</p>  <p>PCS 2302/2024 Laboratório de Fundamentos da Eng.de Computação</p> <p>Aula 04: Introdução ao Paradigma de Objetos</p> <p>Autores: Anna H. R. Costa Jaime S. Sichman João José Neto Paulo S. Muniz Silva Ricardo L. A. Rocha</p> <p>Reestruturação: Paulo S. Muniz Silva</p> <p>v.1.0 ago. 2012</p>	<h2>Exercício 02 :</h2> <h3>Chamada de Supervisor (SVC)</h3>
	<p>TYGXXA05E03 (Obrigatório)</p> <p>Chamada SVC (em Java).</p> <p>Altere o código da MVN para que a partir de uma <i>Chamada de Supervisor</i> (SVC) seja executada a função skip. Essa função avança o cursor de leitura da unidade lógica de disco em uma quantidade específica de bytes.</p> <p>Parâmetro: lu – o número da unidade lógica de I/O num – o número de words a avançar</p> <p>Retorno: número de words avançados, -1 em caso de erro (número da unidade lógica inválida, inexistente ou somente de escrita, ou número negativo de bytes em num)</p> <p>Instrução de Chamada: OS /02FE</p> <p>OBS: O valor do parâmetro a ser passado para a chamada de supervisor deve ser posicionado no endereço de memória anterior ao da chamada.</p> <p>Restrições: Funciona somente se a unidade lógica estiver aberta para leitura e for uma unidade de disco.</p>

DEPARTAMENTO DE ENGENHARIA DE
COMPUTAÇÃO E SISTEMAS DIGITAIS

PCS

PCS 2302/2024
Laboratório de
Fundamentos da
Eng.de Computação

Aula 04:
Introdução ao
Paradigma de
Objetos

Autores:
Anna H. R. Costa
Jaime S. Sichman
João José Neto
Paulo S. Muniz Silva
Ricardo L. A. Rocha



Reestruturação:
Paulo S. Muniz Silva

v.1.0 ago. 2012

Exercício 02 : Chamada de Supervisor (SVC)

Exemplo: reset (para LU=0)	Exemplo: skip (para LU=0, NUM = 2)
JP reset	JP skip
LU K /0000	NUM K /0002
reset OS /01FF	LU K /0000
	skip OS /02FE

Entrega: o código java completo da MVN modificada (conteúdo da pasta src)

DEPARTAMENTO DE ENGENHARIA DE
COMPUTAÇÃO E SISTEMAS DIGITAIS

PCS

PCS 2302/2024
Laboratório de
Fundamentos da
Eng.de Computação

Aula 04:
Introdução ao
Paradigma de
Objetos

Autores:
Anna H. R. Costa
Jaime S. Sichman
João José Neto
Paulo S. Muniz Silva
Ricardo L. A. Rocha

Reestruturação:
Paulo S. Muniz Silva

v.1.0 ago. 2012

Informações auxiliares

Utilize os diagramas e códigos apresentados nas aulas anteriores para lhe ajudar a entender o funcionamento da MVN;

Consulte a documentação das classes da MVN (comentários nos métodos, atributos e classes) para entender o funcionamento da MVN;

Consulte a documentação do Java para lhe auxiliar no desenvolvimento;

O código da MVN disponibilizado funciona para carregar arquivos, executar programas e interagir com a memória. O mesmo deve continuar funcionando após a modificação.

Macro Arquitetura da MVN

