

# Prova P1 – Sistemas de Programação – PCS 3216

Nome: Bernardo Marcelino do Nascimento

Número USP: 9836197

PDF gerado em 13/05/2020 às 17:53

## 1ª Questão

a) Qual é a funcionalidade deste programa?

O código lê os caracteres do vetor iniciado em INIC e escreve esses caracteres na fita de saída.

b) Compatibilize-o com o seu montador e obtenha para ele um código objeto carregável.

O código objeto obtido foi

```
01 00 2d 8e 00 92 01 82 02 91 0c 82 03 91 0d 8e
01 e0 04 82 01 52 00 92 01 11 2a 81 0d 42 00 91
0d 11 22 01 0c 81 0c 42 00 91 0c 01 0c c1 00 6f
00 03 02 00 fb 00 04 02 00 fa 02 00 05 01 00 8e
01 69 0e 00 07 05 74 65 78 74 6f b2
```

O código compatibilizado é

```
      @      100

START LD     COMP
      MM     CONT

IN     LD     LDA0
      MM     LDA
      LD     LDA0+1
      MM     LDA+1

;LDA     LD     INIC
;LDA+1 PD     SAIDA
LDA     K      8e
LDA+1 K      01
      PD     SAIDA

      LD     CONT
      -     UM
      MM     CONT
      JZ     FORA

      LD     LDA+1
      +     UM
      MM     LDA+1

      JZ     INCR
      JP     LDA

INCR   LD     LDA
      +     UM
```

```

        MM    LDA
        JP    LDA

FORA    HM    START

        @    200

UM      K    1
CONT    K    0
;LDA0   LD    INIC
;LDA0+1 K    0
LDA0    K    8e
LDA0+1  K    01

        @    E00
COMP    K    5
INIC    K    74      ; "t"
        K    65      ; "e"
        K    78      ; "x"
        K    74      ; "t"
        K    6F      ; "o"
        #

```

Foram alteradas:

- as linhas com @ pois endereço na máquina não precisa de \ antes
- a instrução LDA para LD
- as linhas que definem LDA e LDA0, pois a implementação do montador não permite o operando com +1
- o endereço da área do vetor de texto, pois a área depois de F00 já era reservada para o loader

c) Teste-o e faça funcionar se encontrar algum erro de lógica

A saída foi

```

74
65
78
74
6f

```

Máquina parada

d) Em lugar de operar sobre a cadeia “texto”, modifique esse programa de modo que passe a operar sobre os dados que você preparou previamente a partir do texto do seu nome (item 2 de preparação para a prova).

Foram alteradas as linhas finais para

```

        @    E00      ; o endereço aqui foi modificado pois F00 é a
localização do loader
COMP    K    10      ; comprimento do texto, em hexa
INIC    K    42      ; "B"          primeira letra do texto
        K    45      ; "E"

```

K	52	;	"R"	
K	4E	;	"N"	
K	41	;	"A"	ultima letra do texto
K	52	;	"R"	
K	44	;	"D"	
K	4F	;	"O"	
K	4D	;	"M"	
K	41	;	"A"	
K	52	;	"R"	
K	43	;	"C"	
K	45	;	"E"	
K	4C	;	"L"	
K	49	;	"I"	
K	4E	;	"N"	ultima letra do texto

e) Teste-o no simulador e comprove seu funcionamento.

### Código Objeto

```
01 00 2d 8e 00 92 01 82 02 91 0c 82 03 91 0d 8e
01 e0 04 82 01 52 00 92 01 11 2a 81 0d 42 00 91
0d 11 22 01 0c 81 0c 42 00 91 0c 01 0c c1 00 6f
00 03 02 00 fb 00 04 02 00 fa 02 00 05 01 00 8e
01 69 0e 00 12 10 42 45 52 4e 41 52 44 4f 4d 41
52 43 45 4c 49 4e 38
```

### Saída do programa

```
42
45
52
4e
41
52
44
4f
4d
41
52
43
45
4c
49
4e
```

Máquina parada

f) Relate e entregue o relatório em arquivo pdf.

## 2ª Questão

O código foi elaborado com base no da questão anterior, com as principais alterações marcadas em negrito. Cada algarismo do número USP é lido do vetor por meio de LDA e é somado a uma variável TOTAL, então faz-se a multiplicação por 16 seguida da divisão por 16, obtendo-se assim o resto da divisão por 16. Esse valor é então escrito na fita de saída e pega-se o próximo algarismo do número USP, repetindo assim o algoritmo até a variável CONT chegar a 0, encerrando o loop.

A saída do programa condiz com o resultado esperado:

7  
0  
1  
7  
a  
2  
b  
b

Máquina parada

O programa feito foi

```
@      497

START LD  COMP
      MM  CONT

IN     LD  LDA0
      MM  LDA
      LD  LDA0+1
      MM  LDA+1

LDA    K    8e
LDA+1  K    01
+      TOTAL ; atualiza TOTAL
MM  TOTAL
*      DEZESSEIS ; multiplica por 0x10 para eliminar o mais significativo
           ; e divide em seguida por 0x10 para ficar so o menos
           ; significativo, que eh o resto da divisao por 16
/      DEZESSEIS
PD     0

; testa se e a ultima letra
      LD  CONT
      -   UM      ; decrementa contador do loop
      MM  CONT
      JZ  FORA    ; se zerar, terminou o loop, se nao, prossegue

; incrementa o endereco a ser acessado no vetor de letras
      LD  LDA+1
      +   UM      ; incrementa byte menos significativo
      MM  LDA+1

      JZ  INCR    ; se deu zero, vai incrementar o outro byte
      JP  LDA     ; se nao, executa mais uma instancia do loop

; incrementa o byte mais significativo do endereco a acessar
INCR  LD  LDA
      +   UM      ; incrementa byte mais significativo
      MM  LDA
      JP  LDA     ; e vai executar outra instancia do loop

; final do loop
```

```

FORA  HM   START   ; para a maquina e volta ao inicio se for reacionada

; ===== aREA DE DADOS =====
;
      @ 200
; constantes e variaveis
UM      K    1      ; constante 1
CONT     K    0      ; variavel CONT iniciada com zero
LDA0     K   8e
LDA0+1   K   01
TOTAL    K    0
DEZESSEIS K  10

      @ E00
COMP K    8
; nusp 09836197
NUSP K   37
      K   39
      K   31
      K   36
      K   33
      K   38
      K   39
      K   30

      #

```

Código objeto:

```

04 97 34 8e 00 92 01 82 02 94 a3 82 03 94 a4 8e
01 42 04 92 04 62 05 72 05 e0 00 82 01 52 00 92
01 14 c9 84 a4 42 00 94 a4 14 c1 04 a3 84 a3 42
00 94 a3 04 a3 c4 97 02 00 06 01 00 8e 01 00 10
0e 00 11 08 37 39 31 36 33 38 39 30 00 00 00 00
00 00 00 00

```

## Memória antes de carregar

[illegible]

[illegible]

[illegible]



```
DEBUG MEM[ef0]: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
DEBUG MEM[f00]: d0 00 9f 44 4f 4b 9f 46 d0 00 9f 45 9f 47 d0 00
DEBUG MEM[f10]: 9f 4a d0 00 0f 46 8f 47 4f 4c 1f 3a 9f 47 8f 4a
DEBUG MEM[f20]: 5f 4c 9f 4a 1f 28 0f 12 d0 00 4f 4b 1f 42 9f 46
DEBUG MEM[f30]: d0 00 9f 47 d0 00 9f 4a 0f 12 8f 46 4f 4c 9f 46
DEBUG MEM[f40]: 0f 1c 30 00 00 00 00 00 0f 16 00 90 01 00 00 00
DEBUG MEM[f50]: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
DEBUG MEM[f60]: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
DEBUG MEM[f70]: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
DEBUG MEM[f80]: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
DEBUG MEM[f90]: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
DEBUG MEM[fa0]: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
DEBUG MEM[fb0]: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
DEBUG MEM[fc0]: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
DEBUG MEM[fd0]: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
DEBUG MEM[fe0]: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
DEBUG MEM[ff0]: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
```

## Memória depois de carregar o programa

[illegible]

[illegible]

[illegible]

```
DEBUG MEM[ef0]: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
DEBUG MEM[f00]: d0 00 9f 44 4f 4b 9f 46 d0 00 9f 45 9f 47 d0 00
DEBUG MEM[f10]: 9f 4a d0 00 0f 46 8f 47 4f 4c 1f 3a 9f 47 8f 4a
DEBUG MEM[f20]: 5f 4c 9f 4a 1f 28 0f 12 d0 00 4f 4b 1f 42 9f 46
DEBUG MEM[f30]: d0 00 9f 47 d0 00 9f 4a 0f 12 8f 46 4f 4c 9f 46
DEBUG MEM[f40]: 0f 1c 30 00 04 97 9e 11 0f 16 00 90 01 00 00 00
DEBUG MEM[f50]: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
DEBUG MEM[f60]: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
DEBUG MEM[f70]: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
DEBUG MEM[f80]: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
DEBUG MEM[f90]: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
DEBUG MEM[fa0]: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
DEBUG MEM[fb0]: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
DEBUG MEM[fc0]: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
DEBUG MEM[fd0]: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
DEBUG MEM[fe0]: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
DEBUG MEM[ff0]: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
```

Após o término

[illegible]

[illegible]

[illegible]



```
DEBUG MEM[ef0]: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
DEBUG MEM[f00]: d0 00 9f 44 4f 4b 9f 46 d0 00 9f 45 9f 47 d0 00
DEBUG MEM[f10]: 9f 4a d0 00 0f 46 8f 47 4f 4c 1f 3a 9f 47 8f 4a
DEBUG MEM[f20]: 5f 4c 9f 4a 1f 28 0f 12 d0 00 4f 4b 1f 42 9f 46
DEBUG MEM[f30]: d0 00 9f 47 d0 00 9f 4a 0f 12 8f 46 4f 4c 9f 46
DEBUG MEM[f40]: 0f 1c 30 00 04 97 9e 11 0f 16 00 90 01 00 00 00
DEBUG MEM[f50]: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
DEBUG MEM[f60]: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
DEBUG MEM[f70]: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
DEBUG MEM[f80]: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
DEBUG MEM[f90]: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
DEBUG MEM[fa0]: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
DEBUG MEM[fb0]: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
DEBUG MEM[fc0]: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
DEBUG MEM[fd0]: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
DEBUG MEM[fe0]: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
DEBUG MEM[ff0]: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
```