**AULA 1!**

1.1 c) 5 -> 0x0000012; 10 -> 0x0000001c; 7 -> 0x00000016

d)

|  |  |  |
| --- | --- | --- |
| Endereço de Memória | Código de Máquina | Instrução |
| 0x00400024 | 0x34080005 | main: ori $t0, $0, 5 |
| 0x00400024 | 0x3408000a | main: ori $t0, $0, 10 |
| 0x00400024 | 0x34080007 | main: ori $t0, $0, 7 |

e)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| PC | Instrução | $t0 | $t1 | $t2 |
| 0x00400024 | ori $t0, $0, 10 | 0x00000000 | 0x00000000 | 0x00000000 |
| 0x00400028 | ori $t2, $0, 8 | 0x0000000a | 0x00000000 | 0x00000000 |
| 0x0040002c | add $t1,$t0, $t0 | 0x0000000a | 0x00000000 | 0x00000008 |
| 0x00400030 | add $t1,$t1,$t2 | 0x0000000a | 0x00000014 | 0x00000008 |
| 0x00400034 | Jr $ra | 0x0000000a | 0x0000001c | 0x00000008 |

2.1 .text

.globl main

main: ori $t0, $0, 10

ori $t2, $0, 8

add $t1, $t0, $t0

sub $t1, $t1, $t2

jr $ra

a)

2 -> 0xffffffe; 3 -> 0xfffffffc; 4 -> 0x00000000; 5 -> 0x00000002

Em 2 e 3, o resultado de $t1 é negativo.

b)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| PC | Instrução | $t0 | $t1 | $t2 |
| 0x00400024 | ori $t0, $0, 3 | 0x00000000 | 0x00000000 | 0x00000000 |
| 0x00400028 | ori $t2, $0, 8 | 0x00000003 | 0x00000000 | 0x00000000 |
| 0x0040002c | add $t1, $t0, $t0 | 0x00000003 | 0x00000000 | 0x00000008 |
| 0x00400030 | sub $t1, $t1, $t2 | 0x00000003 | 0x00000006 | 0x00000008 |
| 0x00400034 | Jr $ra | 0x00000003 | 0xfffffffe | 0x00000008 |

3)

a)

.text

.globl main

main: li $v0, 5

syscall

ori $t2, $0, 8

add $t1, $v0, $v0

sub $a0, $t1, $t2

li $v0, 1

syscall

jr $ra

b)

2 -> -4; 3 -> -2; 5 -> 2

c)

.text

.globl main

main: li $v0, 5

syscall

ori $t2, $0, 8

add $t1, $v0, $v0

sub $a0, $t1, $t2

li $v0, 34

syscall

jr $ra

2 -> 0xfffffffc; 3 -> 0xfffffffe; 4 -> 0x00000000; 5 -> 0x00000002

d)

.text

.globl main

main: li $v0, 5

syscall

ori $t2, $0, 8

add $t1, $v0, $v0

sub $a0, $t1, $t2

li $v0, 36

syscall

jr $ra

2 -> 4294967292; 3 -> 4294967294; 4 -> 0; 5 -> 2

**AULA 2!**

* 1. a)

.text

.globl main

main:

li $t0, “x”

li $t1, “y”

and $t2, $t0, $t1

or $t3, $t0, $t1

nor $t4, $t0, $t1

xor $t5, $t0, $t1

b)

.text

.globl main

main:

li $t0, 0x12345678

li $t1, 0x0000000f

and $t2, $t0, $t1

or $t3, $t0, $t1

nor $t4, $t0, $t1

xor $t5, $t0, $t1

jr $ra

c)

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| $t0 | $t1 | $t2 | $t3 | $t4 | $t5 |
| **0x12345678** | **0x0000000F** | 0x00000008 | 0x1234567f | 0xedcba980 | 0x12345677 |
| 0**x12345678** | **0x000FF000** | 0x00045000 | 0x123ff678 | 0xedc00987 | 0x123ba678 |
| **0x762A5C1B** | **0x89D5A3E4** | 0x00000000 | 0xffffffff | 0x00000000 | 0xffffffff |

d)

.text

.globl main

main:

li $t0, 0x12345678

nor $t1, $t0, $t0

jr $ra

2) b)

.text

.globl main

main: li $t0, 0x12345678

sll $t2, $t0, 1

srl $t3, $t0, 1

sra $t4, $t0, 1

jr $ra

c)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| $t0 | Imm | $t2(sll) | $t3(srl) | $t4(sra) |
| 0x12345678 | 1 | 0x2468acf0 | 0x091a2b3c | 0x091a2b3c |
| 0x12345678 | 4 | 0x23456780 | 0x01234567 | 0x01234567 |
| 0x12345678 | 16 | 0x56780000 | 0x00001234 | 0x00001234 |
| 0x862a5c1b | 2 | 0x18a9706c | 0x218a9706 | 0xe18a9706 |
| 0x862a5c1b | 4 | 0x62a5c1b0 | 0x0862a5c1 | 0xf862a5c1 |

d)

.text

.globl main

main: li $t0, 0x862a5c1b

li $t3, 0xf0000000

and $t1, $t0, $t3

srl $t2, $t1, 28

or $a0, $t2, $0

li $v0, 34

syscall

li $a0, '\n'

li $v0, 11

syscall

li $t3, 0x0f000000

and $t1, $t0, $t3

srl $t2, $t1, 24

or $a0, $t2, $0

li $v0, 34

syscall

li $a0, '\n'

li $v0, 11

syscall

li $t3, 0x00f00000

and $t1, $t0, $t3

srl $t2, $t1, 20

or $a0, $t2, $0

li $v0, 34

syscall

li $a0, '\n'

li $v0, 11

syscall

li $t3, 0x000f0000

and $t1, $t0, $t3

srl $t2, $t1, 16

or $a0, $t2, $0

li $v0, 34

syscall

li $a0, '\n'

li $v0, 11

syscall

li $t3, 0x0000f000

and $t1, $t0, $t3

srl $t2, $t1, 12

or $a0, $t2, $0

li $v0, 34

syscall

li $a0, '\n'

li $v0, 11

syscall

li $t3, 0x00000f00

and $t1, $t0, $t3

srl $t2, $t1, 8

or $a0, $t2, $0

li $v0, 34

syscall

li $a0, '\n'

li $v0, 11

syscall

li $t3, 0x000000f0

and $t1, $t0, $t3

srl $t2, $t1, 4

or $a0, $t2, $0

li $v0, 34

syscall

li $a0, '\n'

li $v0, 11

syscall

li $t3, 0x0000000f

and $t1, $t0, $t3

or $a0, $t2, $0

li $v0, 34

syscall

jr $ra

3)

b)

|  |  |  |  |
| --- | --- | --- | --- |
| Endereço | Valor | Endereço | Valor |
| 0x10010000 | 0x70206f53 | 0x1001001c | 0x74617270 |
| 0x10010004 | 0x20617261 | 0x10010020 | 0x73616369 |
| 0x10010008 | 0x74616863 |  |  |
| 0x1001000c | 0x00726165 |  |  |
| 0x10010010 | 0x20314341 |  |  |
| 0x10010014 | 0x75612013 |  |  |
| 0x10010018 | 0x2073616c |  |  |

c)

.data

str1: .asciiz "So para chatear"

str2: .asciiz "AC1 – aulas praticas"

.text

.globl main

main: la $a0, str2

li $v0, 4

syscall

jr $ra

d)

.data

str1: .asciiz "Introduza 2 numeros: "

str2: .asciiz "A soma dos dois numero é: "

.text

.globl main

main: la $a0, str1

li $v0, 4

syscall

li $v0, 5

syscall

or $t0, $v0, $0

li $v0, 5

syscall

add $t1, $v0, $t0

la $a0, str2

li $v0, 4

syscall

or $a0, $t1, $0

li $v0, 1

syscall

jr $ra

**AULA 3!**

1)

b)

.data

str1: .asciiz "Introduza um número: "

str2: .asciiz "\nO valor em binário: "

.text

.globl main

main: la $a0, str1

li $v0, 4

syscall #imprime str1

li $v0, 5

syscall

ori $t0, $v0, 0 #value = read\_int;

la $a0, str2

li $v0, 4

syscall #imprime str2

li $t1, 0

li $t2, 32

for: beq $t1, $t2, end\_for

andi $t3, $t0, 0x80000000

bne $t3, 0, else

li $a0, '0'

li $v0, 11

syscall

j end\_if

else:

li $a0, '1'

li $v0, 11

syscall

end\_if:

sll $t0, $t0, 1

addiu $t1, $t1, 1

li $t5, 4

divu $t1, $t5

mfhi $t4

bne $t4, 0, end\_if2

li $a0, ' '

li $v0, 11

syscall

end\_if2:

j for

end\_for:

jr $ra

c)

**void main(void)**

**{**

**int value, bit, i;**

**print\_str("Introduza um numero: ");**

**value = read\_int();**

**print\_str("\nO valor em binario: ");**

**for(i=0; i < 32; i++)**

**{**

**bit = value & 0x80000000;**

**bit = bit >> 31;**

**bit = bit + 0x30;**

**print\_char(bit);**

**value = value << 1;**

**}**

**}**

**.data**

**str1: .asciiz "Introduza um número: "**

**str2: .asciiz "\nO valor em binário: "**

**.text**

**.globl main**

**main: la $a0, str1**

**li $v0, 4**

**syscall #imprime str1**

**li $v0, 5**

**syscall**

**ori $t0, $v0, 0 #value = read\_int;**

**la $a0, str2**

**li $v0, 4**

**syscall #imprime str2**

**li $t1, 0**

**for: beq $t1, 32, end\_for**

**andi $t3, $t0, 0x80000000**

**or $t7,$t3,$0**

**srl $t7,$t7,31**

**addi $t7, $t7, 0x30**

**end\_if:**

**addiu $t1, $t1, 1**

**#print char(bit)**

**or $a0, $t7, $0**

**li $v0, 11**

**syscall**

**#criar espaço de 4 em 4**

**li $t5, 4**

**divu $t1, $t5**

**mfhi $t4**

**bne $t4, 0, end\_if2**

**li $a0, ' '**

**li $v0, 11**

**syscall**

**end\_if2:**

**sll $t0, $t0, 1**

**j for**

**end\_for:**

**jr $ra**

**d)**

**void main(void)**

**{**

**int value, bit, i;**

**boolean b = false;**

**print\_str("Introduza um numero: ");**

**value = read\_int();**

**print\_str("\nO valor em binario: ");**

**for(i=0; i < 32; i++)**

**{**

**bit = value & 0x80000000;**

**bit = bit >> 31;**

**bit = bit + 0x30;**

**if(bit == 1)**

**{**

**b = true**

**}**

**if(b)**

**{**

**print\_char(bit);**

**}**

**value = value << 1;**

**}**

**}**

**.data**

**str1: .asciiz "Introduza um número: "**

**str2: .asciiz "\nO valor em binário: "**

**.text**

**.globl main**

**main: la $a0, str1**

**li $v0, 4**

**syscall #imprime str1**

**li $v0, 5**

**syscall**

**ori $t0, $v0, 0 #value = read\_int;**

**la $a0, str2**

**li $v0, 4**

**syscall #imprime str2**

**li $t2, 0**

**li $t1, 0**

**for: beq $t1, 32, end\_for**

**andi $t3, $t0, 0x80000000**

**or $t7,$t3,$0**

**srl $t7,$t7,31**

**addi $t7, $t7, 0x30**

**bne $t7, 0x31, end\_if**

**li $t2, 1**

**end\_if:**

**addiu $t1, $t1, 1**

**bne $t2, 1, end\_if2**

**#print char(bit)**

**or $a0, $t7, $0**

**li $v0, 11**

**syscall**

**#criar espaço de 4 em 4**

**li $t5, 4**

**divu $t1, $t5**

**mfhi $t4**

**bne $t4, 0, end\_if3**

**li $a0, ' '**

**li $v0, 11**

**syscall**

**end\_if3:**

**end\_if2:**

**sll $t0, $t0, 1**

**j for**

**end\_for:**

**jr $ra**

**2)**

**a)**

**.data**

**str1: .asciiz "Introduza um numero: "**

**str2: .asciiz "\nO valor em hexadecimal: "**

**.text**

**.globl main**

**main: la $a0, str1**

**li $v0, 4**

**syscall**

**li $v0, 5**

**syscall**

**or $t0, $v0, $0**

**la $a0, str2**

**li $v0, 4**

**syscall**

**while: andi $t2, $t0, 0xF0000000**

**bne $t2, 0, end\_while**

**blt $t1, 0, end\_while**

**sll $t0, $t0, 4**

**subi $t1, $t1, 1**

**j while**

**end\_while:**

**do:**

**andi $t2, $t0, 0xF0000000**

**srl $t3, $t2, 28**

**addi $a0, $t3, '0'**

**bgt $t3, 9, else**

**li $v0, 11**

**syscall**

**j end\_else**

**else: addi $a0, $a0, 7**

**li $v0, 11**

**syscall**

**end\_else:**

**sll $t0, $t0, 4**

**subi $t1, $t1, 1**

**bgt $t1, 0, do**

**jr $ra**

**b)**

**void main(void)**

**{**

**int value, i, digito;**

**print\_str("Introduza um numero: ");**

**value = read\_int();**

**print\_str("\nO valor em octal: ");**

**i=10;**

**if((value & 0xC0000000) != 0)**

**{**

**digito = (value & 0xC0000000) >> 30;**

**print\_int(digito);**

**}**

**value = value << 2;**

**while( ((value & 0xE0000000) == 0 ) && (i > 0) )**

**{**

**i--;**

**value = value << 3;**

**}**

**do**

**{**

**digito = value >> 29;**

**print\_int(digito);**

**value = (value & 0xE0000000) << 3;**

**i--;**

**} while(i > 0);**

**print\_int(digito);**

**}**

**.data**

**str1: .asciiz "Introduza um numero: "**

**str2: .asciiz "\nO valor em octal: "**

**.text**

**.globl main**

**main: la $a0, str1**

**li $v0, 4**

**syscall**

**li $v0, 5**

**syscall**

**or $t0, $v0, $0**

**la $a0, str2**

**li $v0, 4**

**syscall**

**li $t1, 10**

**if:**

**andi $t2, $t0, 0xC0000000**

**srl $t2, $t2, 30**

**beq $t2, 0, end\_if**

**andi $a0, $t0, 0xC0000000**

**srl $a0, $a0, 30**

**li $v0, 1**

**syscall**

**end\_if:**

**sll $t0, $t0, 2**

**while: andi $t2, $t0, 0xE0000000**

**bne $t2, 0, end\_while**

**beq $t1, 0, end\_while**

**sll $t0, $t0, 3**

**subi $t1, $t1, 1**

**j while**

**end\_while:**

**do:**

**andi $a0, $t0, 0xE0000000**

**srl $a0, $a0, 29**

**li $v0, 1**

**syscall**

**sll $t0, $t0, 3**

**subi $t1, $t1, 1**

**bgt $t1, 0, do**

**jr $ra**

**3)**

**a)**

**.data**

**str1: .asciiz "Introduza dois numeros: "**

**str2: .asciiz "Resultado: "**

**str3: .asciiz "\n"**

**.text**

**.globl main**

**main:**

**#print str1**

**la $a0, str1**

**li $v0, 4**

**syscall**

**#read\_mdor**

**li $v0, 5**

**syscall**

**#$t0 = mdor**

**or $t0, $v0, $0**

**#print "\n"**

**la $a0, str3**

**li $v0, 4**

**syscall**

**#read\_mdo**

**li $v0, 5**

**syscall**

**#$t1 =mdo**

**or $t1, $v0, $0**

**#int i=0**

**li $t2, 0**

**#int res =0**

**li $t5, 0**

**while: beq $t0, 0, end\_while**

**bge $t2, 4, end\_while**

**addi $t2, $t2, 1 #i++**

**if: andi $t3, $t0, 0x00000001 #mdor & 0x00000001**

**beq $t3, 0, end\_if**

**add $t5, $t5, $t1 #res = res + mdo**

**end\_if:**

**sll $t1, $t1, 1**

**srl $t0, $t0, 1**

**j while**

**end\_while:**

**#print str2**

**la $a0, str2**

**li $v0, 4**

**syscall**

**#print res**

**or $a0, $t5, $0**

**li $v0, 1**

**syscall**

**jr $ra**

**b)**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **mdor** | **mdo** | **i** | **res** |  |
| **0x0B** | **0x06** | **0x00** | **0x00** | **Valores iniciais** |
| **0x05** | **0x0c** | **0x01** | **0x06** | **Fim 1ª iteração** |
| **0x02** | **0x18** | **0x02** | **0x12** | **Fim 2ª iteração** |
| **0x01** | **0x30** | **0x03** | **0x12** | **Fim 3ª iteração** |
| **0x00** | **0x60** | **0x04** | **0x42** | **Fim 4ª**  **iteração** |

**c)**

**void main(void)**

**{**

**int res=0, i=0, mdor, mdo;**

**print\_str("Introduza dois numeros: ");**

**mdor = read\_int();**

**mdo = read\_int();**

**while( (mdor != 0) && (i++ < 16) )**

**{**

**if( (mdor & 0x00000001) != 0 )**

**res = res + mdo;**

**mdo = mdo << 1;**

**mdor = mdor >> 1;**

**}**

**print\_str("Resultado: ");**

**print\_int10(res);**

**}**

**.data**

**str1: .asciiz "Introduza dois numeros: "**

**str2: .asciiz "Resultado: "**

**str3: .asciiz "\n"**

**.text**

**.globl main**

**main:**

**#print str1**

**la $a0, str1**

**li $v0, 4**

**syscall**

**#read\_mdor**

**li $v0, 5**

**syscall**

**#$t0 = mdor**

**or $t0, $v0, $0**

**#print "\n"**

**la $a0, str3**

**li $v0, 4**

**syscall**

**#read\_mdo**

**li $v0, 5**

**syscall**

**#$t1 =mdo**

**or $t1, $v0, $0**

**#int i=0($t2)**

**li $t2, 0**

**#int res =0($t5)**

**li $t5, 0**

**while: beq $t0, 0, end\_while**

**bge $t2, 16, end\_while**

**addi $t2, $t2, 1 #i++**

**if: andi $t3, $t0, 0x00000001 #mdor & 0x00000001**

**beq $t3, 0, end\_if**

**add $t5, $t5, $t1 #res = res + mdo**

**end\_if:**

**sll $t1, $t1, 1**

**srl $t0, $t0, 1**

**j while**

**end\_while:**

**#print str2**

**la $a0, str2**

**li $v0, 4**

**syscall**

**#print res**

**or $a0, $t5, $0**

**li $v0, 1**

**syscall**

**jr $ra**

**AULA 4!**

**1) a)**

**.data**

**str: .space 20**

**.text**

**.globl main**

**main:**

**#read\_str**

**la $a0, str**

**li $a1, 20**

**li $v0, 8**

**syscall**

**la $t2, str #$t2=&(str[0])**

**li $t0, 0 #i=0**

**li $t1, 0 #num=0**

**while:**

**addu $t3, $t2, $t0 #$t3 = &(str[i])**

**lb $t4, 0($t3) #$t4= str[i]**

**beq $t4, '\0', end\_while #str[i] != '\0'**

**if: blt $t4, '0', end\_if**

**bgt $t4, '9', end\_if**

**addi $t1, $t1, 1**

**end\_if:**

**addi $t0, $t0, 1**

**j while**

**end\_while:**

**or $a0, $t1, $0**

**li $v0, 1**

**syscall**

**jr $ra**

**b)**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Endereço de str($t2)** | **Endereço de str[i]($t3)** | **Str[i]($t4)** | **n ($t0)** | **num($t1)** |  |
| **0x10010000** | **0x10010000** | **0x00000041** | **0** | **0** | **Val. Inicial** |
| **0x10010000** | **0x10010001** | **0x00000043** | **1** | **0** | **Fim 1ª Iteração** |
| **0x10010000** | **0x10010002** | **0x00000031** | **2** | **1** | **Fim 2ª Iteração** |
| **0x10010000** | **0x10010003** | **0x0000002D** | **3** | **1** | **Fim 3ª Iteração** |
| **0x10010000** | **0x10010004** | **0x00000032** | **4** | **2** | **Fim 4ª Iteração** |
| **0x10010000** | **0x10010005** | **0x00000030** | **5** | **3** | **Fim 5ª Iteração** |
| **0x10010000** | **0x10010006** | **0x00000031** | **6** | **4** | **Fim 6ª Iteração** |
| **0x10010000** | **0x10010007** | **0x00000033** | **7** | **5** | **Fim 7ª Iteração** |
| **0x10010000** | **0x10010008** | **0x0000000a** | **8** | **5** | **Fim 8ª Iteração** |
| **0x10010000** | **0x10010009** | **0x00000000** | **9** | **5** | **Fim** |

**2)**

a)

.data

str: .space 20

.text

.globl main

main: la $a0, str

li $a1, 20

li $v0, 8

syscall #read\_str

li $t0, 0 #$t0(num) = 0

la $t1, str #$t1 = $(str[0])

while:

lb $t2, 0($t1) # \*p =array[i]

beq $t2, '\0', end\_while

if:

blt $t2, '0', end\_if

bgt $t2, '9', end\_if

addi $t0, $t0, 1

end\_if:

addiu $t1, $t1, 1 #p++

j while

end\_while:

or $a0, $t0, $0 #print\_int10(num)

li $v0, 1

syscall

jr $ra

b)

|  |  |  |  |
| --- | --- | --- | --- |
| num($t0) | p($t1) | \*p($t2) |  |
| 0x00000000 | 0x10010000 | 0x00000041 | Valores iniciais |
| 0x00000000 | 0x10010001 | 0x00000041 | Fim 1ª Iteração |
| 0x00000000 | 0x10010002 | 0x00000043 | “ 2ª “ |
| 0x00000001 | 0x10010003 | 0x00000031 | “ 3ª “ |
| 0x00000001 | 0x10010004 | 0x0000002d | “ 4ª “ |
| 0x00000002 | 0x10010005 | 0x00000032 | “ 5ª “ |
| 0x00000003 | 0x10010006 | 0x00000030 | “ 6ª “ |
| 0x00000004 | 0x10010007 | 0x00000031 | “ 7ª “ |
| 0x00000005 | 0x10010008 | 0x00000032 | “ 8ª “ |

3)

a)

.data

str1: .space 20

str2: .asciiz "Introduza uma string: "

.text

.globl main

main: la $a0, str2

li $v0, 4

syscall #print\_str(str2)

la $a0, str1

li $a1, 20

li $v0, 8

syscall #read\_str

la $t0, str1 #$t0 = $(str[0]) - p=str

while: lb $t1, 0($t0) #$t1 = str[i] - \*p = str[i]

beq $t1, '\0', end\_while #while(\*p != '\0')

sub $t1, $t1, 'a' #\*p = \*p - 'a'

add $t1, $t1, 'A' #\*p = \*p + 'A'

sb $t1, 0($t0) #str[i] = \*p

addiu $t0, $t0, 1 #p++

j while

end\_while:

la $a0, str1

li $v0, 4

syscall #print\_str(str2)

jr $ra

b)

|  |  |  |
| --- | --- | --- |
| p($t0) | \*p($t1) |  |
| 0x10010000 | 0x00000061 | Valores Iniciais |
| 0x10010001 | 0x00000041 | Fim 1ª Iteração |
| 0x10010002 | 0x00000043 | “ 2ª “ |
| 0x10010003 | 0x0000002d | “ 3ª “ |
| 0x10010004 | 0x0000002d | “ 4ª “ |
| 0x10010005 | 0x00000050 | “ 5ª “ |
| 0x10010006 | 0x00000052 | “ 6ª “ |
| 0x10010007 | 0x00000041 | “ 7ª “ |
| 0x10010008 | 0x00000054 | “ 8ª “ |
| 0x10010009 | 0x00000049 | “ 9ª “ |
| 0x1001000a | 0x00000043 | “ 10ª “ |
| 0x1001000b | 0x00000041 | “ 11ª “ |
| 0x1001000c | 0x00000053 | “ 12ª “ |

c)

**void main(void)**

**{**

**static char str[20];**

**char \*p;**

**print\_str("Introduza uma string: ");**

**read\_str(str, 20);**

**p = str;**

**while (\*p != '\0')**

**{**

**If(\*p < ‘z’ && \*p > ’a’)**

**\*p = \*p – 'a' + 'A';**

**p++;**

**}**

**print\_str(str);**

**}**

.data

str1: .space 20

str2: .asciiz "Introduza uma string: "

.text

.globl main

main: la $a0, str2

li $v0, 4

syscall #print\_str(str2)

la $a0, str1

li $a1, 20

li $v0, 8

syscall #read\_str

la $t0, str1 #$t0 = $(str[0]) - p=str

while: lb $t1, 0($t0) #$t1 = str[i] - \*p = str[i]

beq $t1, '\0', end\_while #while(\*p != '\0')

bgt $t1, 'z', end\_if

blt $t1, 'a', end\_if

sub $t1, $t1, 'a' #\*p = \*p - 'a'

add $t1, $t1, 'A' #\*p = \*p + 'A'

end\_if:

sb $t1, 0($t0) #str[i] = \*p

addiu $t0, $t0, 1 #p++

j while

end\_while:

la $a0, str1

li $v0, 4

syscall #print\_str(str2)

jr $ra

d)

**void main(void)**

**{**

**static char str[20];**

**char \*p;**

**print\_str("Introduza uma string: ");**

**read\_str(str, 20);**

**p = str;**

**while (\*p != '\0')**

**{**

**If(\*p < ‘Z’ && \*p > ’A’)**

**\*p = \*p – 'A' + 'a';**

**p++;**

**}**

**print\_str(str);**

**}**

.data

str1: .space 20

str2: .asciiz "Introduza uma string: "

.text

.globl main

main: la $a0, str2

li $v0, 4

syscall #print\_str(str2)

la $a0, str1

li $a1, 20

li $v0, 8

syscall #read\_str

la $t0, str1 #$t0 = $(str[0]) - p=str

while: lb $t1, 0($t0) #$t1 = str[i] - \*p = str[i]

beq $t1, '\0', end\_while #while(\*p != '\0')

bgt $t1, 'Z', end\_if

blt $t1, 'A', end\_if

sub $t1, $t1, 'A' #\*p = \*p - 'a'

add $t1, $t1, 'a' #\*p = \*p + 'A'

end\_if: #alínea c)

sb $t1, 0($t0) #str[i] = \*p

addiu $t0, $t0, 1 #p++

j while

end\_while:

la $a0, str1

li $v0, 4

syscall #print\_str(str2)

jr $ra

4)

a)

.data

array: .word 7692, 23, 5, 234

.text

.globl main

main: la $t0, array #p=array

li $t3, 0 #soma=0

la $t1, array #pultimo=array

addiu $t1, $t1, 12 #pultimo = array+3(4\*3)

while: lw $t2, 0($t0) #$t2->\*p, \*p = array[i]

bgt $t0, $t1, end\_while #while(p <= pultimo

add $t3, $t3, $t2 #soma = soma + (\*P)

addiu $t0, $t0, 4 #p++

j while

end\_while:

or $a0, $t3, $0

li $v0, 1

syscall #print\_int10(soma)

jr $ra

b)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| p($t0) | pultimo($t1) | \*p($t2) | soma($t3 |  |
| 0x10010000 | 0x1001000c | 0x00000000 | 0x00000000 | Valores Iniciais |
| 0x10010004 | 0x1001000c | 0x00001e0c | 0x00001e0c | Fim 1ª Iteração |
| 0x10010008 | 0x1001000c | 0x00000017 | 0x00001e23 | “ 2ª “ |
| 0x1001000c | 0x1001000c | 0x00000005 | 0x00001e28 | “ 3ª “ |
| 0x10010010 | 0x1001000c | 0x000000ea | 0x00001f12 | “ 4ª “ |

c)

**int array[] = {7692, 23, 5, 234};**

**void main (void)**

**{**

**int i=0;**

**int soma = 0;**

**while(i<=3)**

**{**

**soma = soma +array[i];**

**i++;**

**}**

**print\_int10(soma);**

**}**

c)

.data

arr: .word 7692, 23, 5, 234

.text

.globl main

main: la $t0, array #$t0 = &(array[0])

li $t1, 0 #i=0

li $t2, 0 #soma =0

while: sll $t3, $t1, 2 #$t3 = i\*4

addu $t3, $t0, $t3 #$t3 = &(array[i])

lw $t4, 0($t3) #$t4 = array[i]

bgt $t1, 3, end\_while #while(i <= 3)

add $t2, $t2, $t4 #soma = soma + array[i]

addi $t1, $t1, 1 #i++

j while

end\_while:

or $a0, $t2, $0

li $v0, 1

syscall #print\_int10(soma)

jr $ra

***AULA 5!***

1)

a)

.data

lista: .space 20

.text

.globl main

main: la $t1, lista # $t1 = $(lista[0])

li $t0, 0 #i =0

for:

sll $t2, $t0, 2 #$t2 =i\*4

addu $t2, $t1, $t2 #$t2 = &(array[i])

beq $t0, 5, end\_for #(i<5)

li $v0, 5

syscall #read\_int()

sw $v0, 0($t2) #array[i] = $v0

addi $t0, $t0, 1 #i++

j for

end\_for:

jr $ra

c)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| i($t0) | lista($t1) | &(lista[i])($t2) | ($v0) | Fim Iteração |
| 0x00000001 | 0x10010000 | 0x10010000 | 0x0000000e | 1ª |
| 0x00000002 | 0x10010000 | 0x10010004 | 0x00001234 | 2ª |
| 0x00000003 | 0x10010000 | 0x10010008 | 0x00ab1256 | 3ª |
| 0x00000004 | 0x10010000 | 0x1001000c | 0xffffffff | 4ª |
| 0x00000005 | 0x10010000 | 0x10010010 | 0xabcd9876 | 5ª |

2)

a)

.data

lista: .word 8, -4, 3, 5, 124, -15, 87, 9, 27, 15

str1: .asciiz “\nConteudo do array:\n”

str2: .asciiz “-“

.text

.globl main

main: la $t0, lista #$t0,p = lista

li $t1, 0

la $a0, str1

li $v0, 4

syscall #printstr(str1)

for: lw $t2, 0($t0) #$t2, \*p = lista[i]

beq $t1, 10, end\_for #i<size

or $a0, $t2, $0

li $v0, 1

syscall #print\_int10(\*p)

la $a0, str2

li $v0, 4

syscall

addu $t0, $t0, 4 #p++

addi $t1, $t1, 1 #i++

j for

end\_for:

jr $ra

**3)**

**a)**

#define SIZE 10

void main(void)

{

static int lista[SIZE];

int houveTroca, i, aux;

// inserir aqui o código para leitura de valores e

p=lista;

for(i=0; i< SIZE;i++)

{

Print\_str(“\nIntroduza um numero: “);

aux = read\_int();

\*p=aux;

p++;

}

// preenchimento do array

do

{

houveTroca = FALSE;

for (i=0; i < SIZE-1; i++)

{

if (lista[i] > lista[i+1])

{

aux = lista[i];

lista[i] = lista[i+1];

lista[i+1] = aux;

houveTroca = TRUE;

}

}

} while (houveTroca==TRUE);

// inserir aqui o código de impressão do conteúdo do array

p=lista;

for(i=0;i<SIZE;i++)

{

print\_int10(lista[i]);

print\_str(“-“);

}

}

**b)**

.data

lista: .space 40

str1: .asciiz "\nIntroduza um numero: "

str2: .asciiz "-"

.text

.globl main

main: la $t0, lista #p=lista

li $t1, 0 #i=0

for1: beq $t1, 10, end\_for1 #i<size

la $a0, str1

li $v0, 4

syscall #print\_str(str1)

li $v0, 5

syscall #read\_int()

sw $v0, 0($t0) # lista[i] = $v0

addiu $t0, $t0, 4 #p++

addi $t1, $t1, 1 #i++

j for1

end\_for1:

do: li $t2, 0 #houveTroca = false

li $t1, 0 #i=0

for2: beq $t1, 9, end\_for2

la $t0, lista

sll $t3, $t1, 2 #$t3=i\*4

addu $t3, $t0, $t3 #$t3 = $(lista[i])

addi $t4, $t1, 1 #$t4=i++

sll $t4, $t4, 2 #$t4 = $t4(i+1)\*4

addu $t4, $t0, $t4 #$t4 = $(lista[i+1])

lw $t5, 0($t3) #$t5 = lista[i]

lw $t6, 0($t4) #$t6 = lista[i+1]

ble $t5, $t6, end\_if #lista[i] > lista [i+1]

sw $t5, 0($t4) #lista[i+1] = lista[i]

sw $t6, 0($t3) #lista[i] = lista[i+1]

li $t2, 1 #houveTroca = true

end\_if:

addi $t1, $t1, 1 #i++

j for2

end\_for2:

bne $t2, 1, end\_while #houvetroca == true

j do

end\_while:

li $t1, 0 #i=0

for3: beq $t1, 10, end\_for3 #i<size

la $t0, lista

sll $t3, $t1, 2 #$t3 = i\*4

addu $t3, $t0, $t3 # $t3 = $(lista[i])

lw $a0, 0($t3) #$a0 = lista[i]

li $v0, 1

syscall #print\_int10(lista[i])

la $a0, str2

li $v0, 4

syscall #print\_str("-")

addi $t1, $t1, 1 #i++

j for3

end\_for3:

jr $ra

**c)**

#define unsigned SIZE 10

void main(void)

{

static int lista[SIZE];

int houveTroca, i, aux;

// inserir aqui o código para leitura de valores e

p=lista;

for(i=0; i< SIZE;i++)

{

Print\_str(“\nIntroduza um numero: “);

aux = read\_int();

\*p=aux;

p++;

}

// preenchimento do array

do

{

houveTroca = FALSE;

for (i=0; i < SIZE-1; i++)

{

if (lista[i] > lista[i+1])

{

aux = lista[i];

lista[i] = lista[i+1];

lista[i+1] = aux;

houveTroca = TRUE;

}

}

} while (houveTroca==TRUE);

// inserir aqui o código de impressão do conteúdo do array

p=lista;

for(i=0;i<SIZE;i++)

{

print\_int10(lista[i]);

print\_str(“-“);

}

}

.data

lista: .space 40

str1: .asciiz "\nIntroduza um numero: "

str2: .asciiz "-"

.text

.globl main

main: la $t0, lista #p=lista

li $t1, 0 #i=0

for1: beq $t1, 10, end\_for1 #i<size

la $a0, str1

li $v0, 4

syscall #print\_str(str1)

li $v0, 5

syscall #read\_int()

sw $v0, 0($t0) # lista[i] = $v0

addiu $t0, $t0, 4 #p++

addi $t1, $t1, 1 #i++

j for1

end\_for1:

do: li $t2, 0 #houveTroca = false

li $t1, 0 #i=0

for2: beq $t1, 9, end\_for2

la $t0, lista

sll $t3, $t1, 2 #$t3=i\*4

addu $t3, $t0, $t3 #$t3 = $(lista[i])

addi $t4, $t1, 1 #$t4=i++

sll $t4, $t4, 2 #$t4 = $t4(i+1)\*4

addu $t4, $t0, $t4 #$t4 = $(lista[i+1])

lw $t5, 0($t3) #$t5 = lista[i]

lw $t6, 0($t4) #$t6 = lista[i+1]

bleu $t5, $t6, end\_if #lista[i] > lista [i+1]

sw $t5, 0($t4) #lista[i+1] = lista[i]

sw $t6, 0($t3) #lista[i] = lista[i+1]

li $t2, 1 #houveTroca = true

end\_if:

addi $t1, $t1, 1 #i++

j for2

end\_for2:

bne $t2, 1, end\_while #houvetroca == true

j do

end\_while:

li $t1, 0 #i=0

for3: beq $t1, 10, end\_for3 #i<size

la $t0, lista

sll $t3, $t1, 2 #$t3 = i\*4

addu $t3, $t0, $t3 # $t3 = $(lista[i])

lw $a0, 0($t3) #$a0 = lista[i]

li $v0, 1

syscall #print\_int10(lista[i])

la $a0, str2

li $v0, 4

syscall #print\_str("-")

addi $t1, $t1, 1 #i++

j for3

end\_for3:

jr $ra

**4)**

**a)**

.data

lista: .space 40

str1: .asciiz "\nInsira um numero: "

str2: .asciiz "-"

.text

.globl main

main: la $t0, lista

li $t1, 0

for1: beq $t1, 10, end\_for1

la $a0, str1

li $v0, 4

syscall

li $v0, 5

syscall

sw $v0, 0($t0)

addiu $t0, $t0, 4

addi $t1, $t1, 1

j for1

end\_for1:

la $t3, lista #$t3->pultimo = lista

addi $t3, $t3, 36 #pultimo = lista+9(4\*9)

do:

la $t0, lista # p=lista

li $t2, 0 # houvetrocas = false

for2: beq $t0, $t3, end\_for2 #p<pultimo

lw $t4, 0($t0) #$t4->\*p = lista[i]

addiu $t5, $t0, 4 #$t5 = p+1

lw $t6, 0($t5) #$t6 = lista[i+1]

ble $t4, $t6, end\_if

sw $t6, 0($t0) #\*p = lista[i+1]

sw $t4, 0($t5) #\*(p+1) = lista[i]

li $t2, 1 #houvetrocas = true

end\_if:

addi $t0, $t0, 4 #p++

j for2

end\_for2:

bne $t2, 1, end\_while

j do

end\_while:

li $t1, 0

for3: beq $t1, 10, end\_for3

la $t0, lista

sll $t2, $t1, 2

addu $t2, $t0, $t2

lw $a0, 0($t2)

li $v0, 1

syscall

la $a0, str2

li $v0, 4

syscall

addi $t1, $t1, 1

j for3

end\_for3:

jr $ra

**5)**

**a)**

.data

lista: .space 40

str1: .asciiz "\nInsira um valor: "

str2: .asciiz "\\"

.text

.globl main

main: la $t0, lista #$t0 = lista[0]

li $t1, 0 #i=0

for1: beq $t1, 10, end\_for1

la $t0, lista

sll $t2, $t1, 2 #i\*4

addu $t2, $t0, $t2 #$t2 = &(lista[i])

la $a0, str1

li $v0, 4

syscall # print\_str(str1)

li $v0, 5

syscall

sw $v0, 0($t2) #lista[i] = $v0

addi $t1, $t1, 1 #i++

j for1

end\_for1:

li $t1, 0 #i=0

for2: beq $t1, 9, end\_for2 #i<size-1

addi $t2, $t1, 1 #j=i+1

for3: beq $t2, 10, end\_for3 #j<size

la $t0, lista #$t0 = lista[0]

sll $t3, $t1, 2 #i\*4

sll $t4, $t2, 2 #j\*4

add $t3, $t0, $t3 # $t3 = &(lista[i])

add $t4, $t0, $t4 # $t4 = &(lista[j])

lw $t5, 0($t3) #$t5 = lista[i]

lw $t6, 0($t4) #$t6 = lista[j]

ble $t5, $t6, end\_if

sw $t5, 0($t4) # lista[j] = lista[i]

sw $t6, 0($t3) # lista[i] = lista[j]

end\_if:

addi $t2, $t2, 1

j for3

end\_for3:

addi $t1, $t1, 1

j for2

end\_for2:

la $t0, lista #$t0, p = lista[0]

li $t1, 0 #i=0

for4: beq $t1, 10, end\_for4 #i<size

lw $a0, 0($t0) #$a0, \*p = lista[i]

li $v0, 1

syscall

la $a0, str2

li $v0, 4

syscall

addi $t0, $t0, 4

addi $t1, $t1, 1

j for4

end\_for4:

jr $ra

**AULA 6!**

**1)**

**a)**

.data

str1: .asciiz "\n"

.text

.globl main

main: move $t0, $a0 #$t0 = size

move $t1, $a1 #$t1 = array[0]

li $t2, 0 #i=0

for: beq $t2, $t0, end\_for

lw $a0, 0($t1)

li $v0, 4

syscall

lb $a0, str1

li $v0, 4

syscall

addu $t1, $t1, 4

addi $t2, $t2, 1

j for

end\_for:

jr $ra

**2)**

**a)**

.data

str1: .asciiz "\n"

.text

.globl main

main: move $t0, $a0

move $t1, $a1

sll $t2, $t0, 2 #size\*4

addu $t2, $t1, $t2 #pultimo = array+size

for:

beq $t1, $t2, end\_for

lw $a0, 0($t1) #$a0=\*p

li $v0, 4

syscall #print\_str(\*p)

la $a0, str1

syscall #print\_str(\n)

addu $t1, $t1, 4 #p++

j for

end\_for:

jr $ra

**3)**

**a)**

.data

str1: .asciiz "\nString #"

str2: .asciiz ": "

str3: .asciiz "-"

.text

.globl main

main:

move $t0, $a0 #$t0 = size

move $t1, $a1 #$t1 = array[0]

li $t2, 0 #i=0

for: beq $t2, $t0, end\_for #i<size

la $a0, str1

li $v0, 4

syscall #print\_str(str1)

move $a0, $t2

li $v0, 1

syscall #print\_int10(i)

la $a0, str2

li $v0, 4

syscall #print\_str(str2)

sll $t3, $t2, 2 #i\*4

add $t3, $t1, $t3 #$t3 = &(array[i])

lw $t4, 0($t3) #$t4 = array[i][0]

li $t5, 0 #j=0

while: add $t6, $t4, $t5 #$t6 = &(array[i][j])

lb $a0, 0($t6) #$a0 = array[i][j]

beq $a0, '\0', end\_while

li $v0, 11

syscall #print\_char(array[i][j])

la $a0, str3

li $v0, 4

syscall #print\_str(str3)

addi $t5, $t5, 1 #j++

j while

end\_while:

addi $t2, $t2, 1

j for

end\_for:

jr $ra

**4)**

**a)**

.data

str1: .asciiz "Nr. de parametros: "

str2: .asciiz "\nP"

str3: .asciiz ". "

.text

.globl main

main: move $t0, $a0 #$t0 = argc = size

move $t1, $a1 #$t1 = argv=&(array[0][0])

li $t2, 0 #i=0

la $a0, str1

li $v0, 4

syscall #print\_str(str1)

or $a0, $t0, $0

li $v0, 1

syscall #print\_int10(argc=size)

for1: beq $t2, $t0, end\_for1

la $a0, str2

li $v0, 4

syscall #print\_str(str2)

or $a0, $t2, $0

li $v0, 1

syscall #printint10(i)

la $a0, str3

li $v0, 4

syscall #print\_str(str3)

sll $t3, $t2, 2 #i\*4

add $t3, $t1, $t3 #$t3= &(array[i][0])

lw $t4, 0($t3) #$t4 = array[i][0]

li $t5, 0 #j=0

for2: add $t6, $t4, $t5 #$t6=&(array[i][j])

lb $a0, 0($t6) #$t6=array[i][j]

beq $a0, '\0', end\_for2

li $v0, 11

syscall #print\_char()

addi $t5, $t5, 1

j for2

end\_for2:

addi $t2, $t2, 1

j for1

end\_for1:

jr $ra

**5)**

**a)**

int main(int argc, char \*argv[])

{

int i, j, max, imax, num, min, mai;

max=0;

imax=0;

for(i=0;i< argc; i++)

{

num=0;

min=0;

mai=0;

print\_str(“\nArgumento: “);

for(j=0;j!= ‘\0’;j++)

{

print\_char(argv[i][j]);

if(argv[i][j]>=‘A’ && argv[i][j]<=‘Z’)

mai++;

else if(argv[i][j] >= ‘a’ && argv[i][j]

<= ‘z’)

min++;

num++;

}

print\_str(“\nNr. Carateres: “);

print\_int(num);

print\_str(“\nMinúsculas: “);

print\_int(min);

print\_str(“\nMaiúsculas: “);

print\_int(mai);

if(num> max)

imax=i;

}

print\_str(“\nMaior STRING: “);

print\_str(arg[imax]);

}

**b)**

.data

str1: .asciiz "\nArgumento: "

str2: .asciiz "\nNr. Carateres: "

str3: .asciiz "\nMinúsculas: "

str4: .asciiz "\nMaiúsculas: "

str5: .asciiz "\nMaior STRING: "

.text

.globl main

main: move $t0, $a0 #$t0=argc=size

move $t1, $a1 #$t1=&(array[0][0])

li $t2, 0 #i=0

li $t3, 0 #max=0

li $t4, 0 #imax=0

for1: beq $t2, $t0, end\_for1

li $t5, 0 #num=0

li $t6, 0 #mai=0

li $t7, 0 #min=0

la $a0, str1

li $v0, 4

syscall #print\_str(str1)

sll $s0, $t2, 2 #i\*4

add $s0, $t1, $s0 #$s0=&(array[i][0])

li $s2, 0 #j=0

for2: lw $s1, 0($s0) #$s1 = array[i][0]

add $s1, $s1, $s2 # $s1=&(array[i][j])

lb $s3, 0($s1) #$s3 = array[i][j]

beq $s3, '\0', end\_for2

or $a0, $s3, $0

li $v0, 11

syscall #print\_char(array[i][j])

blt $s3, 'a', end\_if1

bgt $s3, 'z', end\_if1

addi $t7, $t7, 1 #min++

j end\_if2

end\_if1:blt $s3, 'A', end\_if2

bgt $s3, 'Z', end\_if2

addi $t6, $t6, 1 #mai++

end\_if2:addi $t5, $t5, 1 #num++

addi $s2, $s2, 1 #j++

j for2

end\_for2:

la $a0, str2

li $v0, 4

syscall #print\_str(str2)

or $a0, $t5, $0

li $v0, 1

syscall #print\_int(num)

la $a0, str3

li $v0, 4

syscall #print\_str(str3)

or $a0, $t7, $0

li $v0, 1

syscall #print\_int(min)

la $a0, str4

li $v0, 4

syscall #print\_str(str4)

or $a0, $t6, $0

li $v0, 1

syscall #print\_int(mai)

ble $t5, $t3, end\_if3

or $t4, $t2, $0

or $t3, $t5, $0

end\_if3:

addi $t2, $t2, 1

j for1

end\_for1:

la $a0, str5

li $v0, 4

syscall #print\_str(str5)

li $t2, 0 #i=0

sll $t3, $t4, 2 #imax\*4

add $t3, $t1, $t3 #$t3=&(array[imax][0])

lw $t4, 0($t3) #$t4 = array[imax][0]

for3: add $t5, $t4, $t2 #$t5 = &(array[imax][i])

lb $t6, 0($t5) #$t6 = array[imax][i]

beq $t6, '\0', end\_for3 #array[imax][i] != ‘\0’

or $a0, $t6, $0

li $v0, 11

syscall #print\_char(array[imax[][i])

addi $t2, $t2, 1 #i++

j for3

end\_for3:

jr $ra

**6)**

**a)**

#define SIZE 3

void main(void)

{

static char \*array[SIZE]={"Array", "de", "ponteiros"};

int i;

char \*\*p;

char \*\*pultimo;

p=array;

pultimo = array+SIZE;

for(i=0; p < pultimo; p++)

{

print\_str( "\nString #" );

print\_int10( i );

print\_str( ": " );

while(\*\*p != '\0')

{

print\_char(\*\*p);

print\_char('-');

\*p++;

}

i++;

}

}

**b)**

.data

str1: .asciiz "\nString #"

str2: .asciiz ": "

str3: .asciiz "-"

w1: .asciiz "Array"

w2: .asciiz "de"

w3: .asciiz "Ponteiros"

array: .word w1, w2, w3

.text

.globl main

main: la $t0, array #$t0=p=array

addi $t1, $t0, 12 #$t1 = p+3

li $t2, 0 #i=0

for: beq $t0, $t1, end\_for #p<pultimo

la $a0, str1

li $v0, 4

syscall #print\_str(str1)

or $a0, $t2, $0

li $v0, 1

syscall #print\_int(i)

la $a0, str2

li $v0, 4

syscall #print\_str(str2)

lw $t3, 0($t0) #$t3 = \*p

while: lb $t4, 0($t3) #$t4 = \*\*p

beq $t4, '\0', end\_while

or $a0, $t4, $0

li $v0, 11

syscall #print\_char(\*\*p)

la $a0, str3

li $v0, 4

syscall

addi $t3, $t3, 1 #\*p++

j while

end\_while:

addi $t2, $t2, 1 #i++

addi $t0, $t0, 4 #p++

j for

end\_for:

jr $ra

**AULA 7!**

**1)**

**a)**

strlen: li $t0, 0 #int len=0

while: lb $t1, 0($a0) #char \*s

beq $t1, '\0', end\_while #\*s != '\0'

addi $t0, $t0, 1 #len++

addi $a0, $a0, 1 #\*s++

j while

end\_while:

move $v0, $t0

jr $ra

**b)**

.data

str1: .asciiz "String de teste"

.text

.globl main

main:

la $a0, str1 #\*p = str1

addi $sp,$sp,-4

sw $ra,0($sp)

jal strlen

lw $ra,0($sp)

addi $sp,$sp, 4

move $a0, $v0

li $v0, 1

syscall #print\_int10(strlen(str1))

jr $ra

**c)**

.data

str1: .asciiz " - "

.text

.globl main

main:

move $t0, $a0, #$t0 = argc

move $t1, $a1 #$t1 = \*argv[]

li $t2, 0 #i=0

while1: bge $t2, $t0, end\_while1

li $a0, '\n'

li $v0, 11

syscall #print\_char('\n')

lw $a0, 0($t1) #$a0 = argv[i]

li $v0, 4

syscall #print\_str(argv[i])

la $a0, str1

li $v0, 4

syscall #print\_str(" - ")

lw $a0, 0($t1) #$a0 = argv[i]

addi $sp, $sp, -16

sw $ra, 0($sp)

sw $t0, 4($sp)

sw $t1, 8($sp)

sw $t2, 12($sp)

jal strlen

lw $ra, 0($sp)

lw $t0, 4($sp)

lw $t1, 8($sp)

lw $t2, 12($sp)

addi $sp, $sp, 16

move $a0, $v0

li $v0, 1

syscall #print\_int10(strlen(argv[i]))

addi $t2, $t2, 1 #i++

addi $t1, $t1, 4 #\*argv[]++

j while1

end\_while1:

jr $ra

strlen: li $t0, 0 #int len=0

while: lb $t1, 0($a0) #char \*s

beq $t1, '\0', end\_while #\*s != '\0'

addi $t0, $t0, 1 #len++

addi $a0, $a0, 1 #\*s++

j while

end\_while:

move $v0, $t0

jr $ra

**2)**

.data

str1: .asciiz "ortauQ ed \*\*\* aut A"

.text

.globl main

main:

la $t0, str1 #\*p = str1

move $a0, $t0

addi $sp,$sp,-4

sw $ra,0($sp)

jal strrev

lw $ra,0($sp)

addi $sp,$sp, 4

move $a0, $v0

li $v0, 4

syscall #print\_str(strrev(str1))

jr $ra

strrev: la $t1, 0($a0) #\*p1 = str

la $t2, 0($a0) #\*p2 = str

move $t0, $a0 #$t0 = &(str)

while1: lb $t3, 0($t2) #$t2 = p2

beq $t3, '\0', end\_while1

addi $t2, $t2, 1 #\*p2++

j while1

end\_while1:

addi $t2, $t2, -1 #\*p2--

while2: bge $t1, $t2, end\_while2

move $a0, $t1 #$a0 = p1

move $a1, $t2 #$a1 = p2

#inicio guardar e chamar exchange

addiu $sp, $sp, -16

sw $ra, 0($sp)

sw $t0, 4($sp)

sw $t1, 8($sp)

sw $t2, 12($sp)

jal exchange

lw $ra, 0($sp)

lw $t0, 4($sp)

lw $t1, 8($sp)

lw $t2, 12($sp)

addiu $sp, $sp, 16

#fim de carregar e chamar exchange

addi $t1, $t1, 1 #\*p1++

addi $t2, $t2, -1 #\*p2--

j while2

end\_while2:

move $v0, $t0

jr $ra

exchange:

lb $t0, 0($a0) #aux = \*c1

lb $t1, 0($a1) #aux2 = \*c2

sb $t0, 0($a1) #\*c1=\*c2

sb $t1, 0($a0) #\*c2 = aux

jr $ra

**3)**

**a)**

strcpy:

move $t0, $a0 #$t0 = \*dst

li $t2, 0 #i=0

while2: add $t3, $t2, $a1 #$t3=&(src[i])

add $t4, $t2, $a0 #$t4=&(dst[i])

lb $t5, 0($t3) #$t3=src[i]

sb $t5, 0($t4) #dst[i] = src[i]

beq $t5, '\0', end\_while2

addi $t2, $t2, 1 #i++

j while2

end\_while2:

move $v0, $t0

jr $ra

**b)**

.data

str: .space 11

str2: .asciiz "String too long. Nothing done!\n"

.text

.globl main

main:

move $t0, $a0 #$t0 = argc

move $t1, $a1 #$t1 = \*argv[]

bne $t0, 1, end #if(argc ==1)

lw $a0, 0($t1) #$a0 = argv[0]

#

addiu $sp, $sp, -12

sw $ra, 0($sp)

sw $t0, 4($sp)

sw $t1, 8($sp)

jal strlen

lw $ra, 0($sp)

lw $t0, 4($sp)

lw $t1, 8($sp)

addiu $sp, $sp, 12

#

move $t2, $v0 #$t2 = strlen(argv[0])

bgt $t2, 10, else #if(strlen(argv[i]) <= STR\_MAX\_SIZE)

la $a0, str #$a0 = &(str(buf))

lw $a1, 0($t1) #$a1 = &(argv[0])

#

addiu $sp, $sp, -4

sw $ra, 0($sp)

jal strcpy

lw $ra, 0($sp)

addiu $sp, $sp, 4

#

move $a0, $v0

li $v0, 4

syscall #print\_str(str(buf))

j end

else:

la $a0, str2

li $v0, 4

syscall

end:

jr $ra

strlen: li $t0, 0 #int len=0

while1: lb $t1, 0($a0) #char \*s

beq $t1, '\0', end\_while1 #\*s != '\0'

addi $t0, $t0, 1 #len++

addi $a0, $a0, 1 #\*s++

j while1

end\_while1:

move $v0, $t0 #return len

jr $ra

strcpy:

move $t0, $a0 #$t0 = \*dst

li $t2, 0 #i=0

while2: add $t3, $t2, $a1 #$t3=&(src[i])

add $t4, $t2, $a0 #$t4=&(dst[i])

lb $t5, 0($t3) #$t3=src[i]

sb $t5, 0($t4) #dst[i] = src[i]

beq $t5, '\0', end\_while2

addi $t2, $t2, 1 #i++

j while2

end\_while2:

move $v0, $t0

jr $ra

**c)**

char \*strcpy(char \*dst, char \*src)

{

Int i=0;

do

{

\*dst = \*src;

dst++;

} while(src++ != ‘\0’);

Return dst;

}

.data

str: .space 11

str2: .asciiz "String too long. Nothing done!\n"

.text

.globl main

main:

move $t0, $a0 #$t0 = argc

move $t1, $a1 #$t1 = \*argv[]

bne $t0, 1, end #if(argc ==1)

lw $a0, 0($t1) #$a0 = argv[0]

#

addiu $sp, $sp, -12

sw $ra, 0($sp)

sw $t0, 4($sp)

sw $t1, 8($sp)

jal strlen

lw $ra, 0($sp)

lw $t0, 4($sp)

lw $t1, 8($sp)

addiu $sp, $sp, 12

#

move $t2, $v0 #$t2 = strlen(argv[0])

bgt $t2, 10, else #if(strlen(argv[i]) <= STR\_MAX\_SIZE)

la $a0, str #$a0 = &(str(buf))

lw $a1, 0($t1) #$a1 = &(argv[0])

#

addiu $sp, $sp, -4

sw $ra, 0($sp)

jal strcpy

lw $ra, 0($sp)

addiu $sp, $sp, 4

#

move $a0, $v0

li $v0, 4

syscall #print\_str(str(buf))

j end

else:

la $a0, str2

li $v0, 4

syscall

end:

jr $ra

strlen: li $t0, 0 #int len=0

while1: lb $t1, 0($a0) #char \*s

beq $t1, '\0', end\_while1 #\*s != '\0'

addi $t0, $t0, 1 #len++

addi $a0, $a0, 1 #\*s++

j while1

end\_while1:

move $v0, $t0 #return len

jr $ra

strcpy:

move $t0, $a0 #$t0 = dst

li $t1, 0 #i=0

while:

lb $t2, 0($a1) #$t2=\*src

sb $t2, 0($a0) #\*dst=\*src

beq $t2, '\0', end\_while

addi $a0, $a0, 1 #dst++

addi $a1, $a1, 1 #src++

j while

end\_while:

move $v0, $t0 #return dst

jr $ra

**4)**

**a)**

strcat: move $t0, $a0 #$t0 = &(dst)

while: lb $t1, 0($a0) #$t1=\*p

beq $t1, '\0', end\_while

addi $a0, $a0, 1 #p++

j while

end\_while:

#

addi $sp, $sp, -8

sw $ra, 0($sp)

sw $t0, 4($sp)

jal strcpy

lw $ra, 0($sp)

lw $t0, 4($sp)

addi $sp, $sp, 8

#

move $v0, $t0

jr $ra

**b)**

.data

str1: .asciiz "Arquitetura de "

str2: .space 50

str3: .asciiz "Computadores"

.text

.globl main

main: la $a1, str1

la $a0, str2

#

addi $sp, $sp, -4

sw $ra, 0($sp)

jal strcpy

lw $ra, 0($sp)

addi $sp, $sp, 4

#

move $a0, $v0

la $a1, str3

#

addi $sp, $sp, -4

sw $ra, 0($sp)

jal strcat

lw $ra, 0($sp)

addi $sp, $sp, 4

#

move $a0, $v0

li $v0, 4

syscall

jr $ra

strcat: move $t0, $a0 #$t0 = &(dst)

while: lb $t1, 0($a0) #$t1=\*p

beq $t1, '\0', end\_while

addi $a0, $a0, 1 #p++

j while

end\_while:

#

addi $sp, $sp, -8

sw $ra, 0($sp)

sw $t0, 4($sp)

jal strcpy

lw $ra, 0($sp)

lw $t0, 4($sp)

addi $sp, $sp, 8

#

move $v0, $t0

jr $ra

strcpy:

move $t0, $a0 #$t0 = \*dst

li $t2, 0 #i=0

while2: add $t3, $t2, $a1 #$t3=&(src[i])

add $t4, $t2, $a0 #$t4=&(dst[i])

lb $t5, 0($t3) #$t3=src[i]

sb $t5, 0($t4) #dst[i] = src[i]

beq $t5, '\0', end\_while2

addi $t2, $t2, 1 #i++

j while2

end\_while2:

move $v0, $t0

jr $ra

**AULA 8!**

**1)**

**a)**

atoi: li $t0, 0 #res=0

while: lb $t1, 0($a0) #$t1=\*s

blt $t1, '0', end\_while

bgt $t1, '9', end\_while

subi $t2, $t1, '0' #digit = \*s -'0'

addi $a0, $a0, 1 #\*s++

mulo $t0, $t0, 10 #res = res\*10

add $t0, $t0, $t2 #res = res+digit

j while

end\_while:

move $v0, $t0

**b)**

.data

str1: .asciiz "2040, o ano do fim das PPP"

.text

.globl main

main: la $a0, str1

#

addi $sp, $sp, -4

sw $ra, 0($sp)

jal atoi

lw $ra, 0($sp)

addi $sp, $sp, 4

#

move $a0, $v0

li $v0, 1

syscall

jr $ra

atoi: li $t0, 0 #res=0

while: lb $t1, 0($a0) #$t1=\*s

blt $t1, '0', end\_while

bgt $t1, '9', end\_while

subi $t2, $t1, '0' #digit = \*s -'0'

addi $a0, $a0, 1 #\*s++

mulo $t0, $t0, 10 #res = res\*10

add $t0, $t0, $t2 #res = res+digit

j while

end\_while:

move $v0, $t0

jr $ra

**c)**

unsigned int atoi(char \*s)

{

Unsigned int digit=0, bin, num=0, total=0, res =0;

While((\*s >= ‘0’) && (\*s <= ‘1’))

{

Total++;

S++;

}

s--;

While(total > 0)

{

Bin = \*s-- - ‘0’;

Bin = bin<<num++;

res = res+bin;

Total--;

}

Return res;

}

.data

str1: .asciiz "11111111"

.text

.globl main

main: la $a0, str1

#

addi $sp, $sp, -4

sw $ra, 0($sp)

jal atoi

lw $ra, 0($sp)

addi $sp, $sp, 4

#

move $a0, $v0

li $v0, 1

syscall

jr $ra

atoi:

li $t0, 0 #res=0

li $t1, 0 #total=0

while1: lb $t2, 0($a0)

blt $t2, '0', end\_while1

bgt $t2, '1', end\_while1

addi $t1, $t1, 1 #total++

addi $a0, $a0, 1 #s++

j while1

end\_while1:

addi $a0, $a0, -1 #s--

li $t3, 0 #num=0

while2: ble $t1, 0, end\_while2

lb $t2, 0($a0)

subi $t5, $t2, '0'

addi $a0, $a0, -1 #s--

sllv $t5, $t5, $t3 #bin<<num

addi $t3, $t3, 1 #num++

add $t0, $t0, $t5 #res=res+bin

addi $t1, $t1, -1

j while2

end\_while2:

move $v0, $t0

jr $ra

**2)**

**b)**

int main(void)

{

Static char str[32];

Int num, base;

Print\_str(“Número: “);

Num = read\_int();

Print\_Str(“Base: “);

Base = read\_int();

Print\_str(itoa(num, base, str));

Return 0;

}

.data

str1: .space 32

str2: .asciiz "Número: "

str3: .asciiz "Base: "

.text

.globl main

main: la $a0, str2

li $v0, 4

syscall #print\_str(NUmero)

li $v0, 5

syscall #read\_int(num)

move $t0, $v0

la $a0, str3

li $v0, 4

syscall #print\_str(Base)

li $v0, 5

syscall #read\_int(base)

move $a1, $v0

move $a0, $t0

la $a2, str1

#itoa(n, b str1)

#

addi $sp, $sp, -4

sw $ra, 0($sp)

jal itoa

lw $ra, 0($sp)

addi $sp, $sp, 4

#

move $a0, $v0

li $v0, 4

syscall

jr $ra

itoa:

move $t0, $a2

move $t1, $a0 #$t1=n

move $t2, $a1 #$t2=b

while: div $t1, $t2 #n%b

mfhi $t3 #digit = n%b

mflo $t1 #n=n/b

move $a0, $t3 #toascii(digit)

#

addi $sp, $sp, -20

sw $ra, 0($sp)

sw $t0, 4($sp)

sw $t1, 8($sp)

sw $t2, 12($sp)

sw $a2, 16($sp)

jal toascii

lw $ra, 0($sp)

lw $t0, 4($sp)

lw $t1, 8($sp)

lw $t2, 12($sp)

lw $a2, 16($sp)

addi $sp, $sp, 20

#

sb $v0, 0($a2) #\*p=toascii(digit)

addi $a2, $a2, 1 #p++

ble $t1, 0, end\_while

j while

end\_while:

li $t1, '\0'

sb $t1, 0($a2) #\*p = '\0'

move $a0, $t0 #strrev(s)

addi $sp, $sp, -8

sw $ra, 0($sp)

sw $t0, 4($sp)

jal strrev

lw $ra, 0($sp)

lw $t0, 4($sp)

addi $sp, $sp, 8

#

move $v0, $t0

jr $ra

toascii:

addi $a0, $a0, '0'

ble $a0, '9', end\_if

addi $a0, $a0, 7

end\_if: move $v0, $a0

jr $ra

strrev: la $t1, 0($a0) #\*p1 = str

la $t2, 0($a0) #\*p2 = str

move $t0, $a0 #$t0 = &(str)

while1: lb $t3, 0($t2) #$t2 = p2

beq $t3, '\0', end\_while1

addi $t2, $t2, 1 #\*p2++

j while1

end\_while1:

addi $t2, $t2, -1 #\*p2--

while2: bge $t1, $t2, end\_while2

move $a0, $t1 #$a0 = p1

move $a1, $t2 #$a1 = p2

#inicio guardar e chamar exchange

addiu $sp, $sp, -16

sw $ra, 0($sp)

sw $t0, 4($sp)

sw $t1, 8($sp)

sw $t2, 12($sp)

jal exchange

lw $ra, 0($sp)

lw $t0, 4($sp)

lw $t1, 8($sp)

lw $t2, 12($sp)

addiu $sp, $sp, 16

#fim de carregar e chamar exchange

addi $t1, $t1, 1 #\*p1++

addi $t2, $t2, -1 #\*p2--

j while2

end\_while2:

move $v0, $t0

jr $ra

exchange:

lb $t0, 0($a0) #aux = \*c1

lb $t1, 0($a1) #aux2 = \*c2

sb $t0, 0($a1) #\*c1=\*c2

sb $t1, 0($a0) #\*c2 = aux

jr $ra

**c)**

.data

buf: .space 33

str1: .asciiz "Número: "

str2: .asciiz "Base: "

.text

.globl main

main:

la $a0, str1

li $v0, 4

syscall #print\_str(NUmero)

li $v0, 5

syscall #read\_int(num)

move $t0, $v0

la $a0, str2

li $v0, 4

syscall #print\_str(Base)

li $v0, 5

syscall #read\_int(base)

move $a1, $v0

move $a0, $t0

#

addi $sp, $sp, -4

sw $ra, 0($sp)

jal print\_int\_ac1

lw $ra, 0($sp)

addi $sp, $sp, 4

#

jr $ra

print\_int\_ac1:

la $a2, buf

#

addi $sp, $sp, -4

sw $ra, 0($sp)

jal itoa

lw $ra, 0($sp)

addi $sp, $sp, 4

#

move $a0, $v0

li $v0, 4

syscall

jr $ra

itoa:

move $t0, $a2

move $t1, $a0 #$t1=n

move $t2, $a1 #$t2=b

while: div $t1, $t2 #n%b

mfhi $t3 #digit = n%b

mflo $t1 #n=n/b

move $a0, $t3 #toascii(digit)

#

addi $sp, $sp, -20

sw $ra, 0($sp)

sw $t0, 4($sp)

sw $t1, 8($sp)

sw $t2, 12($sp)

sw $a2, 16($sp)

jal toascii

lw $ra, 0($sp)

lw $t0, 4($sp)

lw $t1, 8($sp)

lw $t2, 12($sp)

lw $a2, 16($sp)

addi $sp, $sp, 20

#

sb $v0, 0($a2) #\*p=toascii(digit)

addi $a2, $a2, 1 #p++

ble $t1, 0, end\_while

j while

end\_while:

li $t1, '\0'

sb $t1, 0($a2) #\*p = '\0'

move $a0, $t0 #strrev(s)

addi $sp, $sp, -8

sw $ra, 0($sp)

sw $t0, 4($sp)

jal strrev

lw $ra, 0($sp)

lw $t0, 4($sp)

addi $sp, $sp, 8

#

move $v0, $t0

jr $ra

toascii:

addi $a0, $a0, '0'

ble $a0, '9', end\_if

addi $a0, $a0, 7

end\_if: move $v0, $a0

jr $ra

strrev: la $t1, 0($a0) #\*p1 = str

la $t2, 0($a0) #\*p2 = str

move $t0, $a0 #$t0 = &(str)

while1: lb $t3, 0($t2) #$t2 = p2

beq $t3, '\0', end\_while1

addi $t2, $t2, 1 #\*p2++

j while1

end\_while1:

addi $t2, $t2, -1 #\*p2--

while2: bge $t1, $t2, end\_while2

move $a0, $t1 #$a0 = p1

move $a1, $t2 #$a1 = p2

#inicio guardar e chamar exchange

addiu $sp, $sp, -16

sw $ra, 0($sp)

sw $t0, 4($sp)

sw $t1, 8($sp)

sw $t2, 12($sp)

jal exchange

lw $ra, 0($sp)

lw $t0, 4($sp)

lw $t1, 8($sp)

lw $t2, 12($sp)

addiu $sp, $sp, 16

#fim de carregar e chamar exchange

addi $t1, $t1, 1 #\*p1++

addi $t2, $t2, -1 #\*p2--

j while2

end\_while2:

move $v0, $t0

jr $ra

exchange:

lb $t0, 0($a0) #aux = \*c1

lb $t1, 0($a1) #aux2 = \*c2

sb $t0, 0($a1) #\*c1=\*c2

sb $t1, 0($a0) #\*c2 = aux

jr $ra

**3)**

**a)**

.data

str1: .asciiz "\nOperação desconhecida"

str2: .asciiz "\nNúmero de argumentos errado"

buf: .space 33

.text

.globl main

main: move $t0, $a0 #$t0=argc

move $t1, $a1 #$t1=argv

li $t2, 0 #exit\_code=0

bne $t0, 3, end\_if

lw $a0, 0($t1) #$a0=argv[0]

#

addi $sp, $sp, -12

sw $ra, 0($sp)

sw $t1, 4($sp)

sw $t2, 8($sp)

jal atoi

lw $ra, 0($sp)

lw $t1, 4($sp)

lw $t2, 8($sp)

addi $sp, $sp, 12

#

move $t3, $v0 #$t3=val1=atoi(argv[0])

lw $a0, 8($t1) #$a0=argv[2]

#

addi $sp, $sp, -16

sw $ra, 0($sp)

sw $t1, 4($sp)

sw $t2, 8($sp)

sw $t3, 12($sp)

jal atoi

lw $ra, 0($sp)

lw $t1, 4($sp)

lw $t2, 8($sp)

lw $t3, 12($sp)

addi $sp, $sp, 16

#

move $t4, $v0 #$t4=val2=atoi(argv[2])

lw $t5, 4($t1) #$t5=&argv[1][0]

lb $t7, 0($t5) #$t7=op=argv[1][0]

bne $t7, 'x', end\_if1 #op=='x'

mulo $t6, $t3, $t4 # res = val1 \* val2

j end\_if4

end\_if1:

bne $t7, '/', end\_if2

div $t6, $t3, $t4 #res = val1 / val2

j end\_if4

end\_if2:

bne $t7, '%', end\_if3

div $t3, $t4

mfhi $t6 #res = val1 % val2

j end\_if4

end\_if3:

la $a0, str1

li $v0, 4

syscall #print\_str(str1)

li $t2, 2 #exit\_code=2

end\_if4:

j if

end\_if:

la $a0, str2

li $v0, 4

syscall

li $t2, 1 #exit\_code=1

if:

bne $t2, 0, end\_if5

move $a0, $t3

li $a1, 10

#

addi $sp, $sp, -16

sw $ra, 0($sp)

sw $t4, 4($sp)

sw $t7, 8($sp)

sw $t6, 12($sp)

jal print\_int\_ac1

lw $ra, 0($sp)

lw $t4, 4($sp)

lw $t7, 8($sp)

lw $t6, 12($sp)

addi $sp, $sp, 16

#

move $a0, $t7

li $v0, 11

syscall #print\_char(op)

move $a0, $t4

li $a1, 10

#

addi $sp, $sp, -8

sw $ra, 0($sp)

sw $t6, 4($sp)

jal print\_int\_ac1

lw $ra, 0($sp)

lw $t6, 4($sp)

addi $sp, $sp, 8

#

li $a0, '='

li $v0, 11

syscall #print\_char('=')

move $a0, $t6

li $a1, 10

#

addi $sp, $sp, -4

sw $ra, 0($sp)

jal print\_int\_ac1

lw $ra, 0($sp)

addi $sp, $sp, 4

#

end\_if5:

jr $ra

atoi: li $t0, 0 #res=0

while: lb $t1, 0($a0) #$t1=\*s

blt $t1, '0', end\_while

bgt $t1, '9', end\_while

subi $t2, $t1, '0' #digit = \*s -'0'

addi $a0, $a0, 1 #\*s++

mulo $t0, $t0, 10 #res = res\*10

add $t0, $t0, $t2 #res = res+digit

j while

end\_while:

move $v0, $t0

jr $ra

print\_int\_ac1:

la $a2, buf

#

addi $sp, $sp, -4

sw $ra, 0($sp)

jal itoa

lw $ra, 0($sp)

addi $sp, $sp, 4

#

move $a0, $v0

li $v0, 4

syscall

jr $ra

itoa:

move $t0, $a2

move $t1, $a0 #$t1=n

move $t2, $a1 #$t2=b

while3: div $t1, $t2 #n%b

mfhi $t3 #digit = n%b

mflo $t1 #n=n/b

move $a0, $t3 #toascii(digit)

#

addi $sp, $sp, -20

sw $ra, 0($sp)

sw $t0, 4($sp)

sw $t1, 8($sp)

sw $t2, 12($sp)

sw $a2, 16($sp)

jal toascii

lw $ra, 0($sp)

lw $t0, 4($sp)

lw $t1, 8($sp)

lw $t2, 12($sp)

lw $a2, 16($sp)

addi $sp, $sp, 20

#

sb $v0, 0($a2) #\*p=toascii(digit)

addi $a2, $a2, 1 #p++

ble $t1, 0, end\_while3

j while3

end\_while3:

li $t1, '\0'

sb $t1, 0($a2) #\*p = '\0'

move $a0, $t0 #strrev(s)

addi $sp, $sp, -8

sw $ra, 0($sp)

sw $t0, 4($sp)

jal strrev

lw $ra, 0($sp)

lw $t0, 4($sp)

addi $sp, $sp, 8

#

move $v0, $t0

jr $ra

toascii:

addi $a0, $a0, '0'

ble $a0, '9', end\_if23

addi $a0, $a0, 7

end\_if23: move $v0, $a0

jr $ra

strrev: la $t1, 0($a0) #\*p1 = str

la $t2, 0($a0) #\*p2 = str

move $t0, $a0 #$t0 = &(str)

while1: lb $t3, 0($t2) #$t2 = p2

beq $t3, '\0', end\_while1

addi $t2, $t2, 1 #\*p2++

j while1

end\_while1:

addi $t2, $t2, -1 #\*p2--

while2: bge $t1, $t2, end\_while2

move $a0, $t1 #$a0 = p1

move $a1, $t2 #$a1 = p2

#inicio guardar e chamar exchange

addiu $sp, $sp, -16

sw $ra, 0($sp)

sw $t0, 4($sp)

sw $t1, 8($sp)

sw $t2, 12($sp)

jal exchange

lw $ra, 0($sp)

lw $t0, 4($sp)

lw $t1, 8($sp)

lw $t2, 12($sp)

addiu $sp, $sp, 16

#fim de carregar e chamar exchange

addi $t1, $t1, 1 #\*p1++

addi $t2, $t2, -1 #\*p2--

j while2

end\_while2:

move $v0, $t0

jr $ra

exchange:

lb $t0, 0($a0) #aux = \*c1

lb $t1, 0($a1) #aux2 = \*c2

sb $t0, 0($a1) #\*c1=\*c2

sb $t1, 0($a0) #\*c2 = aux

jr $ra

**4)**

**a)**

.data

.text

.globl main

main: li $a0, 120

li $a1, 10

addi $sp, $sp, -4

sw $ra, 0($sp)

jal div

lw $ra, 0($sp)

addi $sp, $sp, 4

move $t0, $v0

srl $t1, $t0, 16 # return>>17

beq $t1, $0, end

move $a0, $t1

li $v0, 1

syscall

j end2

end:

move $a0, $t0

li $v0, 1

syscall

end2:

jr $ra

div: sll $t1, $a1, 16 #divisor <<16

andi $t0, $a0, 0xFFFF #dividendo & oxFFFF

sll $t0, $t0, 1 #dividendo<<1

li $t2, 0 #i=0

for: bge $t2, 16, end\_for

li $t3, 0 #bit=0

blt $t0, $t1, end\_if123 #dividendo >= divisor

sub $t0, $t0, $t1 #dividendo - divisor

li $t3, 1 #bit=1

end\_if123:

sll $t0, $t0, 1 #dividendo <<1

or $t0, $t0, $t3 #dividendo | bit

addi $t2, $t2, 1 #i++

j for

end\_for:

srl $t4, $t0, 1 #resto = dividendo >> 1

andi $t4, $t4, 0xFFFF0000 # resto & 0xFFFF0000

andi $t5, $t0, 0xFFFF # dividendo & 0xFFFF

or $v0, $t4, $t5 #return(resto | quociente)

jr $ra

**b)**

int main(int argc, char \*argv[])

{

int val1, val2, res, exit\_code, aux;

char op;

exit\_code = 0;

if(argc == 3)

{

val1 = atoi(argv[0]);

val2 = atoi(argv[2]);

op = argv[1][0];

if(op == 'x')

res = val1 \* val2;

else if(op == '/')

res = div(val1, val2);

res = res<<16;

res = res>>16;

else if(op == '%')

{

res = div(val1, val2);

res=res>>16;

}

else

{

print\_str("\nOperacao desconhecida");

exit\_code = 1;

}

}

else

{

print\_str("\nNumero de argumentos errado");

exit\_code = 2;

}

if(exit\_code == 0)

{

print\_int\_ac1(val1, 10);

print\_char(op);

print\_int\_ac1(val2, 10);

print\_char('=');

print\_int\_ac1(res, 10);

}

return exit\_code;

}

.data

str1: .asciiz "\nOperação desconhecida"

str2: .asciiz "\nNúmero de argumentos errado"

buf: .space 33

.text

.globl main

main: move $t0, $a0 #$t0=argc

move $t1, $a1 #$t1=argv

li $t2, 0 #exit\_code=0

bne $t0, 3, end\_if

lw $a0, 0($t1) #$a0=argv[0]

#

addi $sp, $sp, -12

sw $ra, 0($sp)

sw $t1, 4($sp)

sw $t2, 8($sp)

jal atoi

lw $ra, 0($sp)

lw $t1, 4($sp)

lw $t2, 8($sp)

addi $sp, $sp, 12

#

move $t3, $v0 #$t3=val1=atoi(argv[0])

lw $a0, 8($t1) #$a0=argv[2]

#

addi $sp, $sp, -16

sw $ra, 0($sp)

sw $t1, 4($sp)

sw $t2, 8($sp)

sw $t3, 12($sp)

jal atoi

lw $ra, 0($sp)

lw $t1, 4($sp)

lw $t2, 8($sp)

lw $t3, 12($sp)

addi $sp, $sp, 16

#

move $t4, $v0 #$t4=val2=atoi(argv[2])

lw $t5, 4($t1) #$t5=&argv[1][0]

lb $t7, 0($t5) #$t7=op=argv[1][0]

bne $t7, 'x', end\_if1 #op=='x'

mulo $t6, $t3, $t4 # res = val1 \* val2

j end\_if4

end\_if1:

bne $t7, '/', end\_if2

move $a0, $t3

move $a1, $t4

#

addi $sp, $sp, -24

sw $ra, 0($sp)

sw $t2, 4($sp)

sw $t3, 8($sp)

sw $t4, 12($sp)

sw $t5, 16($sp)

sw $t7, 20($sp)

jal div

lw $ra, 0($sp)

lw $t2, 4($sp)

lw $t3, 8($sp)

lw $t4, 12($sp)

lw $t5, 16($sp)

lw $t7, 20($sp)

addi $sp, $sp, 24

#

sll $v0, $v0, 16 #res<<16

srl $v0, $v0, 16 #res>>16

move $t6, $v0 #res = val1 / val2

j end\_if4

end\_if2:

bne $t7, '%', end\_if3

move $a0, $t3

move $a1, $t4

#

addi $sp, $sp, -24

sw $ra, 0($sp)

sw $t2, 4($sp)

sw $t3, 8($sp)

sw $t4, 12($sp)

sw $t5, 16($sp)

sw $t7, 20($sp)

jal div

lw $ra, 0($sp)

lw $t2, 4($sp)

lw $t3, 8($sp)

lw $t4, 12($sp)

lw $t5, 16($sp)

lw $t7, 20($sp)

addi $sp, $sp, 24

#

srl $v0, $v0, 16 #res>>17

move $t6, $v0 #res = val1 % val2

j end\_if4

end\_if3:

la $a0, str1

li $v0, 4

syscall #print\_str(str1)

li $t2, 2 #exit\_code=2

end\_if4:

j if

end\_if:

la $a0, str2

li $v0, 4

syscall

li $t2, 1 #exit\_code=1

if:

bne $t2, 0, end\_if5

move $a0, $t3

li $a1, 10

#

addi $sp, $sp, -16

sw $ra, 0($sp)

sw $t4, 4($sp)

sw $t7, 8($sp)

sw $t6, 12($sp)

jal print\_int\_ac1

lw $ra, 0($sp)

lw $t4, 4($sp)

lw $t7, 8($sp)

lw $t6, 12($sp)

addi $sp, $sp, 16

#

move $a0, $t7

li $v0, 11

syscall #print\_char(op)

move $a0, $t4

li $a1, 10

#

addi $sp, $sp, -8

sw $ra, 0($sp)

sw $t6, 4($sp)

jal print\_int\_ac1

lw $ra, 0($sp)

lw $t6, 4($sp)

addi $sp, $sp, 8

#

li $a0, '='

li $v0, 11

syscall #print\_char('=')

move $a0, $t6

li $a1, 10

#

addi $sp, $sp, -4

sw $ra, 0($sp)

jal print\_int\_ac1

lw $ra, 0($sp)

addi $sp, $sp, 4

#

end\_if5:

jr $ra

atoi: li $t0, 0 #res=0

while: lb $t1, 0($a0) #$t1=\*s

blt $t1, '0', end\_while

bgt $t1, '9', end\_while

subi $t2, $t1, '0' #digit = \*s -'0'

addi $a0, $a0, 1 #\*s++

mulo $t0, $t0, 10 #res = res\*10

add $t0, $t0, $t2 #res = res+digit

j while

end\_while:

move $v0, $t0

jr $ra

print\_int\_ac1:

la $a2, buf

#

addi $sp, $sp, -4

sw $ra, 0($sp)

jal itoa

lw $ra, 0($sp)

addi $sp, $sp, 4

#

move $a0, $v0

li $v0, 4

syscall

jr $ra

itoa:

move $t0, $a2

move $t1, $a0 #$t1=n

move $t2, $a1 #$t2=b

while3: div $t1, $t2 #n%b

mfhi $t3 #digit = n%b

mflo $t1 #n=n/b

move $a0, $t3 #toascii(digit)

#

addi $sp, $sp, -20

sw $ra, 0($sp)

sw $t0, 4($sp)

sw $t1, 8($sp)

sw $t2, 12($sp)

sw $a2, 16($sp)

jal toascii

lw $ra, 0($sp)

lw $t0, 4($sp)

lw $t1, 8($sp)

lw $t2, 12($sp)

lw $a2, 16($sp)

addi $sp, $sp, 20

#

sb $v0, 0($a2) #\*p=toascii(digit)

addi $a2, $a2, 1 #p++

ble $t1, 0, end\_while3

j while3

end\_while3:

li $t1, '\0'

sb $t1, 0($a2) #\*p = '\0'

move $a0, $t0 #strrev(s)

addi $sp, $sp, -8

sw $ra, 0($sp)

sw $t0, 4($sp)

jal strrev

lw $ra, 0($sp)

lw $t0, 4($sp)

addi $sp, $sp, 8

#

move $v0, $t0

jr $ra

toascii:

addi $a0, $a0, '0'

ble $a0, '9', end\_if23

addi $a0, $a0, 7

end\_if23: move $v0, $a0

jr $ra

strrev: la $t1, 0($a0) #\*p1 = str

la $t2, 0($a0) #\*p2 = str

move $t0, $a0 #$t0 = &(str)

while4: lb $t3, 0($t2) #$t2 = p2

beq $t3, '\0', end\_while4

addi $t2, $t2, 1 #\*p2++

j while4

end\_while4:

addi $t2, $t2, -1 #\*p2--

while5: bge $t1, $t2, end\_while5

lb $t4, 0($t1) #$t3=p1

lb $t5, 0($t2) #$t4=p2

move $a0, $t4 #$a0 = p1

move $a1, $t5 #$a1 = p2

#inicio guardar e chamar exchange

addi $sp, $sp, -8

sw $ra, 0($sp)

sw $t0, 4($sp)

jal exchange

lw $ra, 0($sp)

lw $t0, 4($sp)

addi $sp, $sp, 8

#fim de carregar e chaamr exchange

addi $t1, $t1, 1 #\*p1++

addi $t2, $t2, -1 #\*p2--

j while5

end\_while5:

move $v0, $t0

jr $ra

exchange:

lb $t0, 0($a0) #aux = \*c1

lb $t1, 0($a1) #aux2 = \*c2

sb $t0, 0($a1) #\*c1=\*c2

sb $t1, 0($a0) #\*c2 = aux

jr $ra

div: sll $t1, $a1, 16 #divisor <<16

andi $t0, $a0, 0xFFFF #dividendo & oxFFFF

sll $t0, $t0, 1 #dividendo<<1

li $t2, 0 #i=0

for: bge $t2, 16, end\_for

li $t3, 0 #bit=0

blt $t0, $t1, end\_if123 #dividendo >= divisor

sub $t0, $t0, $t1 #dividendo - divisor

li $t3, 1 #bit=1

end\_if123:

sll $t0, $t0, 1 #dividendo <<1

or $t0, $t0, $t3 #dividendo | bit

addi $t2, $t2, 1 #i++

j for

end\_for:

srl $t4, $t0, 1 #resto = dividendo >> 1

andi $t4, $t4, 0xFFFF0000 # resto & 0xFFFF0000

andi $t5, $t0, 0xFFFF # dividendo & 0xFFFF

or $v0, $t4, $t5 #return(resto | quociente)

jr $ra

**AULA 9!**

**1)**

**a)**

int main(int argc, char \*argv[])

{

If(argc==1)

{

Print\_int(strlen(argv[0]);

Return 0;

}

Return 1;

}

**b)**

.data

.text

.globl main

main: bne $a0, 1, end

lw $a0, 0($a1) #$a0=argv[0]

#

addi $sp, $sp, -4

sw $ra, 0($sp)

jal strlen

lw $ra, 0($sp)

addi $sp, $sp, 4

#

move $a0, $v0

li $v0, 1

syscall

end: jr $ra

strlen:

lb $t0, 0($a0)

beq $t0, '\0', else

addi $a0, $a0, 1 #s+1

#

addi $sp, $sp, -4

sw $ra, 0($sp)

jal strlen

lw $ra, 0($sp)

addi $sp, $sp, 4

#

addi $v0, $v0, 1 #return(1+strlen(s+1))

j end2

else: li $v0, 0 #return 0

end2: jr $ra

**2)**

**a)**

int main(int argc, char \*argv[])

{

Static char buff[50];

If(argc==1)

{

Print\_str(strcpy(buuf, argv[0]);

Return 0;

}

Return 1;

}

**b)**

.data

buff: .space 50

.text

.globl main

main: bne $a0, 1, end

la $a0, buff #dst=buff

lw $a1, 0($a1) #src=argv[0]

#

addi $sp, $sp, -4

sw $ra, 0($sp)

jal strcpy

lw $ra, 0($sp)

addi $sp, $sp, 4

#

move $a0, $v0

li $v0, 4

syscall

end: jr $ra

strcpy:

move $t0, $a0 #$t0=dst

lb $t1, 0($a1) #$t1=\*src

sb $t1, 0($a0) #\*dst=$t1=\*src

beq $t1, '\0', end\_if

addi $a0, $a0, 1 #dst+1

addi $a1, $a1, 1 #src+1

#

addi $sp, $sp, -8

sw $ra, 0($sp)

sw $t0, 4($sp)

jal strcpy

lw $ra, 0($sp)

lw $t0, 4($sp)

addi $sp, $sp, 8

#

end\_if: move $v0, $t0

jr $ra

**3)**

**b)**

int array[]={321, 25, 34, 2};

static final int SIZE=4;

void main(void)

{

Print\_int(soma(array, SIZE));

}

.data

array: .word 321, 25, 34, 2

.text

.globl main

main: la $a0, array #$a0=array

li $a1, 4 #$a1=SIZE=3

#

addi $sp, $sp, -4

sw $ra, 0($sp)

jal soma

lw $ra, 0($sp)

addi $sp, $sp, 4

#

move $a0, $v0

li $v0, 1

syscall

jr $ra

soma: beq $a1, 0, else

lw $t0, 0($a0)#$t0=\*array

addi $a0, $a0, 4 #array+1

addi $a1, $a1, -1 #nelem-1

#

addi $sp, $sp, -8

sw $ra, 0($sp)

sw $t0, 4($sp)

jal soma

lw $ra, 0($sp)

lw $t0, 4($sp)

addi $sp, $sp, 8

#

add $v0, $v0, $t0 #\*array+soma(array+1, nelem-1)

j end

else:

li $v0, 0

end: jr $ra

**4)**

**a)**

void main(void)

{

Unsigned Int num, base;

Num=?;

Base=?;

Print\_int\_ac1(num, base);

}

**b)**

.data

.text

.globl main

main: li $a0, 435

li $a1, 16

#

addi $sp, $sp, -4

sw $ra, 0($sp)

jal print\_int\_ac1

lw $ra, 0($sp)

addi $sp, $sp, 4

#

jr $ra

print\_int\_ac1:

div $a0, $a1

mfhi $t0

mflo $a0

beq $a0, 0, end #if(num/base)

#

addi $sp, $sp, -8

sw $ra, 0($sp)

sw $t0, 4($sp)

jal print\_int\_ac1

lw $ra, 0($sp)

lw $t0, 4($sp)

addi $sp, $sp, 8

#

end:

move $a0, $t0 #toascii(num % base)

#

addi $sp, $sp, -4

sw $ra, 0($sp)

jal toascii

lw $ra, 0($sp)

addi $sp, $sp, 4

#

move $a0, $v0

li $v0, 11

syscall #print\_char

jr $ra

toascii:

addi $t0, $a0, '0' #v+= '0'

ble $t0, '9', end\_if

addi $t0, $t0, 7

end\_if:

move $v0, $t0

jr $ra

**5)**

**a)**

void main(void)

{

Unsigned Int n;

N=?;

Print\_int(fact(n));

}

**b)**

.data

.text

.globl main

main: li $a0, 4

#

addi $sp, $sp, -4

sw $ra, 0($sp)

jal fact

lw $ra, 0($sp)

addi $sp, $sp, 4

#

move $a0, $v0

li $v0, 1

syscall

jr $ra

fact:

blt $a0, 12, else

li $v0, 0

j end

else:

ble $a0, 1, else2

move $t0, $a0

addi $a0, $a0, -1 #n-1

#

addi $sp, $sp, -8

sw $ra, 0($sp)

sw $t0, 4($sp)

jal fact

lw $ra, 0($sp)

lw $t0, 4($sp)

addi $sp, $sp, 8

#

mulo $v0, $t0, $v0#n\*fact(n-1)

j end

else2:

li $v0, 1#: 1

end: jr $ra

**6)**

**a)**

void main(void)

{

Int x, y;

X=?;

Y=?;

Print\_int(xtoy(x, y));

}

**b)**

.data

.text

.globl main

main: li $a0, 2

li $a1, 9

#

addi $sp, $sp, -4

sw $ra, 0($sp)

jal xtoy

lw $ra, 0($sp)

addi $sp, $sp, 4

#

move $a0, $v0

li $v0, 1

syscall

jr $ra

xtoy: beq $a1, 0, else

move $t0, $a0

addi $a1, $a1, -1 #y-1

#

addi $sp, $sp, -8

sw $ra, 0($sp)

sw $t0, 4($sp)

jal xtoy

lw $ra, 0($sp)

lw $t0, 4($sp)

addi $sp, $sp, 8

#

mulo $v0, $t0, $v0 #x\*xtoy(x, y-1)

j end

else:

li $v0, 1

end: jr $ra

**7)**

**a)**

.data

str1: .asciiz "\nIntroduza o numero de discos: "

str2: .asciiz "\n"

str3: .asciiz " - Mover disco de topo de "

str4: .asciiz " para "

.text

.globl main

main: la $a0, str1

li $v0, 4

syscall #print\_str(str1)

li $v0, 5

syscall #read\_int

move $a0, $v0 #ndiscs=$a0=read\_int()

ble $a0, 0, end1

li $a1, 1

li $a2, 3

li $a3, 2

#

addi $sp, $sp, -4

sw $ra, 0($sp)

jal tohanoi

lw $ra, 0($sp)

addi $sp, $sp, 4

#

end1: jr $ra

tohanoi:

beq $a0, 1, else

move $t1, $a0 #n

move $t2, $a1 #p1

move $t3, $a2 #p2

move $t4, $a3 #p3

addi $a0, $t1, -1 #n-1

move $a2, $t4 #$a2=p3

move $a3, $t3 #$a3=p2

#

addi $sp, $sp, -20

sw $ra, 0($sp)

sw $t1, 4($sp)

sw $t2, 8($sp)

sw $t3, 12($sp)

sw $t4, 16($sp)

jal tohanoi

lw $ra, 0($sp)

lw $t1, 4($sp)

lw $t2, 8($sp)

lw $t3, 12($sp)

lw $t4, 16($sp)

addi $sp, $sp, 20

#

move $a0, $t2

move $a1, $t3

addi $t0, $t0, 1

move $a2, $t0

#

addi $sp, $sp, -20

sw $ra, 0($sp)

sw $t1, 4($sp)

sw $t2, 8($sp)

sw $t3, 12($sp)

sw $t4, 16($sp)

jal print\_msg

lw $ra, 0($sp)

lw $t1, 4($sp)

lw $t2, 8($sp)

lw $t3, 12($sp)

lw $t4, 16($sp)

addi $sp, $sp, 20

#

addi $a0, $t1, -1 #n-1

move $a1, $t4 #$a1=p3

move $a2, $t3 #$a2=p2

move $a3, $t2 #$a3=p1

#

addi $sp, $sp, -20

sw $ra, 0($sp)

sw $t1, 4($sp)

sw $t2, 8($sp)

sw $t3, 12($sp)

sw $t4, 16($sp)

jal tohanoi

lw $ra, 0($sp)

lw $t1, 4($sp)

lw $t2, 8($sp)

lw $t3, 12($sp)

lw $t4, 16($sp)

addi $sp, $sp, 20

#

j end23

else:

move $a0, $a1

move $a1, $a2

addi $t0, $t0,1

move $a2, $t0

#

addi $sp, $sp, -4

sw $ra, 0($sp)

jal print\_msg

lw $ra, 0($sp)

addi $sp, $sp, 4

#

end23: jr $ra

print\_msg:

move $t1, $a0 #t1

move $t2, $a1 #t2

move $t3, $a2 #cnt

la $a0, str2

li $v0, 4

syscall #print\_str("\n")

move $a0, $t3

li $a1, 10

#

addi $sp, $sp, -16

sw $ra, 0($sp)

sw $t1, 4($sp)

sw $t2, 8($sp)

sw $t3, 12($sp)

jal print\_int\_ac1

lw $ra, 0($sp)

lw $t1, 4($sp)

lw $t2, 8($sp)

lw $t3, 12($sp)

addi $sp, $sp, 16

#

la $a0, str3

li $v0, 4

syscall #print\_str(" - Mover disco de topo de ")

move $a0, $t1

li $a1, 10

#

addi $sp, $sp, -8

sw $ra, 0($sp)

sw $t2, 4($sp)

jal print\_int\_ac1

lw $ra, 0($sp)

lw $t2, 4($sp)

addi $sp, $sp, 8

#

la $a0, str4

li $v0, 4

syscall #print\_str(" para ")

move $a0, $t2

li $a1, 10

#

addi $sp, $sp, -4

sw $ra, 0($sp)

jal print\_int\_ac1

lw $ra, 0($sp)

addi $sp, $sp, 4

#

jr $ra

print\_int\_ac1:

div $a0, $a1

mfhi $t1

mflo $a0

beq $a0, 0, end #if(num/base)

#

addi $sp, $sp, -8

sw $ra, 0($sp)

sw $t1, 4($sp)

jal print\_int\_ac1

lw $ra, 0($sp)

lw $t1, 4($sp)

addi $sp, $sp, 8

#

end:

move $a0, $t1 #toascii(num % base)

#

addi $sp, $sp, -4

sw $ra, 0($sp)

jal toascii

lw $ra, 0($sp)

addi $sp, $sp, 4

#

move $a0, $v0

li $v0, 11

syscall #print\_char

jr $ra

toascii:

addi $t1, $a0, '0' #v+= '0'

ble $t1, '9', end\_if

addi $t1, $t1, 7

end\_if:

move $v0, $t1

jr $ra

**AULA 10!**

**1)**

**a)**

.data

val1: .float 2.59375

aux: .float 0.0

.text

.globl main

main:

do: li $v0, 5

syscall #read\_int

move $t0, $v0

mtc1 $t0, $f0 #$f0=val

cvt.s.w $f0, $f0 #(float) val

l.s $f2, val1 #$f2=2.59375

mul.s $f12, $f0, $f2 #$f12=res = (floar) val \* 2.59375

li $v0, 2

syscall #prin\_float(res)

l.s $f4, aux

c.eq.s $f12, $f4

bc1t do

jr $ra

**b)**

0100|0000|1111|1001|0000|0000|0000|0000

4 0 F 9 0 0 0 0

**2)**

**a)**

double f2c(double ft)

{

Double c;

C= (double)5/9 \*(ft-32);

Return c;

}

**b)**

void main(void)

{

Double f, c;

Print\_str(“Insira a tempratura em farenheit: “);

F=Read\_double();

C=f2c(f);

Print\_double(c);

}

.data

str1: .asciiz "Insira a temperatura em farenheit: "

.text

.globl main

main: la $a0, str1

li $v0, 4

syscall #print\_str

li $v0, 7

syscall #read\_double()

mov.d $f12, $f0 #$f12 = f

#

addi $sp, $sp, -4

sw $ra, 0($sp)

jal f2c

lw $ra, 0($sp)

addi $sp, $sp, 4

#

mov.d $f12, $f0

li $v0, 3

syscall

jr $ra

f2c:

li $t0, 5

mtc1 $t0, $f2

cvt.d.w $f2, $f2 #(double) 5

li $t0, 9

mtc1 $t0, $f4

cvt.d.w $f4, $f4 #(double) 9

li $f6, 32

mtc1 $t0, $f6

cvt.d.w $f6, $f6 #(double) 32

div.d $f8, $f2, $f4 #$f8 =5/9

sub.d $f10, $f12, $f6 #$f10 = ft-32

mul.d $f8, $f8, $f10 #5/9 \* (ft-32)

mov.d $f0, $f8

jr $ra

**3)**

**a)**

double average(double \*array, unsigned int n)

{

Double soma=0, media, aux=n;

While(aux>0)

{

Soma +=\*array++;

Aux--;

}

Media = soma/n;

Return media;

}

Void main(void)

{

Static double array[11];

double \*aux;

double media;

Unsigned int n=11, i=0;

While(i<11)

{

Print\_str(“Valor: “);

Array[i++]=read\_double();

}

Aux=array;

Media=average(aux, n);

Print\_double(media);

}

**b)**

.data

str1: .asciiz "Valor: "

array: .space 88

soma: .double 0.0

.text

.globl main

main: li $t0, 0 #i=0

la $t1, str1

la $t2, array

while1: bge $t0, 11, end\_while1

la $a0, str1

li $v0, 4

syscall #print\_str

li $v0, 7

syscall #read\_double

sll $t3, $t0, 3 #i\*8

add $t3, $t2, $t3 #$t3=&(array[i])

s.d $f0, 0($t3) #array[i] = read\_double()

addi $t0, $t0, 1 #i++

j while1

end\_while1:

move $a0, $t2

li $a1, 11

#

addi $sp, $sp, -4

sw $ra, 0($sp)

jal average

lw $ra, 0($sp)

addi $sp, $sp, 4

#

mov.d $f12, $f0

li $v0, 3

syscall

jr $ra

average:

l.d $f4, soma #$f4=soma

move $t0, $a1 #$t0=aux=n

while: ble $t0, 0, end\_while #aux>0

l.d $f6, 0($a0) #$f6=\*array

add.d $f4, $f4, $f6 #soma = soma + \*array

addi $a0, $a0, 8 #array++

addi $t0, $t0, -1 #aux--

j while

end\_while:

mtc1 $a1, $f8

cvt.d.w $f8, $f8 #(double) n

div.d $f0, $f4, $f8 #soma/n

jr $ra

**4)**

**a)**

double max(double \*array, unsigned int n)

{

Double max=0, aux=n;

While(aux>0)

{

If(max < \*array)

Max=\*array;

Array++;

Aux--;

}

Return max;

}

**b)**

Void main(void)

{

Static double array[11];

double \*aux;

double media, max;

Unsigned int n=11, i=0;

While(i<11)

{

Print\_str(“Valor: “);

Array[i++]=read\_double();

}

Aux=array;

Media=average(aux, n);

Print\_double(media);

Max = max(aux, n);

Print\_double(max);

}

.data

str1: .asciiz "Valor: "

array: .space 88

aux: .double 0.0

.text

.globl main

main: li $t0, 0 #i=0

la $t2, array

while1: bge $t0, 11, end\_while1

la $a0, str1

li $v0, 4

syscall #print\_str(str1)

li $v0, 7

syscall #read\_double

sll $t3, $t0, 3 #i\*8

add $t3, $t2, $t3 #$t3=&(array[i])

s.d $f0, 0($t3) #array[i] = read\_double()

addi $t0, $t0, 1 #i++

j while1

end\_while1:

move $a0, $t2

li $a1, 11

#

addi $sp, $sp, -8

sw $ra, 0($sp)

sw $t2, 4($sp)

jal average

lw $ra, 0($sp)

lw $t2, 4($sp)

addi $sp, $sp, 8

#

mov.d $f12, $f0

li $v0, 3

syscall #print\_double(media)

li $a0, '\n'

li $v0, 11

syscall #print\_str("\n")

move $a0, $t2

li $a1, 11

#

addi $sp, $sp, -4

sw $ra, 0($sp)

jal max

lw $ra, 0($sp)

addi $sp, $sp, 4

#

mov.d $f12, $f0

li $v0, 3

syscall #print\_double(max)

jr $ra

average:

l.d $f4, aux #$f4=soma

move $t0, $a1 #$t0=aux=n

while: ble $t0, 0, end\_while #aux>0

l.d $f6, 0($a0) #$f6=\*array

add.d $f4, $f4, $f6 #soma = soma + \*array

addi $a0, $a0, 8 #array++

addi $t0, $t0, -1 #aux--

j while

end\_while:

mtc1 $a1, $f8

cvt.d.w $f8, $f8 #(double) n

div.d $f0, $f4, $f8 #soma/n

jr $ra

max: l.d $f0, aux #max=0

move $t0, $a1 #aux=n

while2: ble $t0, 0, end\_while2

l.d $f6, 0($a0) #$f6=\*array

c.lt.d $f0, $f6

bc1f end\_if

mov.d $f0, $f6 #max = \*array

end\_if: addi $a0, $a0, 8 #array++

addi $t0, $t0, -1 #aux--

j while2

end\_while2:

jr $ra

**5)**

**a)**

.data

str1: .asciiz "Valor: "

array: .space 88

aux: .double 0.0

.text

.globl main

main: li $t0, 0 #i=0

la $t2, array

while1: bge $t0, 11, end\_while1

la $a0, str1

li $v0, 4

syscall #print\_str(str1)

li $v0, 7

syscall #read\_double

sll $t3, $t0, 3 #i\*8

add $t3, $t2, $t3 #$t3=&(array[i])

s.d $f0, 0($t3) #array[i] = read\_double()

addi $t0, $t0, 1 #i++

j while1

end\_while1:

move $a0, $t2

li $a1, 11

#

addi $sp, $sp, -8

sw $ra, 0($sp)

sw $t2, 4($sp)

jal average

lw $ra, 0($sp)

lw $t2, 4($sp)

addi $sp, $sp, 8

#

mov.d $f12, $f0

li $v0, 3

syscall #print\_double(media)

li $a0, '\n'

li $v0, 11

syscall #print\_str("\n")

move $a0, $t2

li $a1, 11

#

addi $sp, $sp, -8

sw $ra, 0($sp)

sw $t2, 4($sp)

jal max

lw $ra, 0($sp)

lw $t2, 4($sp)

addi $sp, $sp, 8

#

mov.d $f12, $f0

li $v0, 3

syscall #print\_double(max)

li $a0, '\n'

li $v0, 11

syscall #print\_str("\n")

move $a0, $t2

li $a1, 11

#

addi $sp, $sp, -8

sw $ra, 0($sp)

sw $t2, 0($sp)

jal sort

lw $ra, 0($sp)

lw $t2, 0($sp)

addi $sp, $sp, 8

#

mov.d $f12, $f0

li $v0, 3

syscall #print\_double(mediana)

li $a0, '\n'

li $v0, 11

syscall #print\_str("\n")

li $t0, 0

for12: beq $t0, 11, end\_for12

sll $t1, $t0, 3 #i\*8

add $t1, $t1, $t2

l.d $f12, 0($t1) #$f2=array[i]

li $v0, 3

syscall #print\_double(array[i])

li $a0, '\n'

li $v0, 11

syscall #print\_str("\n")

addi $t0, $t0, 1 #i++

j for12

end\_for12:

jr $ra

average:

l.d $f4, aux #$f4=soma

move $t0, $a1 #$t0=aux=n

while: ble $t0, 0, end\_while #aux>0

l.d $f6, 0($a0) #$f6=\*array

add.d $f4, $f4, $f6 #soma = soma + \*array

addi $a0, $a0, 8 #array++

addi $t0, $t0, -1 #aux--

j while

end\_while:

mtc1 $a1, $f8

cvt.d.w $f8, $f8 #(double) n

div.d $f0, $f4, $f8 #soma/n

jr $ra

max: l.d $f0, aux #max=0

move $t0, $a1 #aux=n

while2: ble $t0, 0, end\_while2

l.d $f6, 0($a0) #$f6=\*array

c.lt.d $f0, $f6

bc1f end\_if

mov.d $f0, $f6 #max = \*array

end\_if: addi $a0, $a0, 8 #array++

addi $t0, $t0, -1 #aux--

j while2

end\_while2:

jr $ra

sort:

move $t0, $a0 #$t0=\*array

move $t1, $a1 #$t1=nval

do32: li $t2, 0 #houveTroca=false(0)

li $t3, 0 #i=0

addi $t4, $t1, -1 #nval-1

for32: bge $t3, $t4, end\_for32

sll $t5, $t3, 3 #i\*8

add $t5, $t0, $t5 #$t5= &(array[i])

l.d $f0, 0($t5) #$f0=array[i]

add $t6, $t3, 1 #i+1

sll $t6, $t6, 3 #(i+1)\*8

add $t6, $t0, $t6 #$t5= &(array[i+1])

l.d $f2, 0($t6) #$f2=array[i+1]

c.le.d $f0, $f2

bc1t end\_if32 #array[i] > array[i+1]

mov.d $f4, $f0 #aux=array[i]

s.d $f2, 0($t5) #array[i] = array[i+1]

s.d $f4, 0($t6) #array[i+1] = aux(array[i])

li $t2, 1 #houveTrocas=1

end\_if32:

addi $t3, $t3, 1 #i++

j for32

end\_for32:

beq $t2, 1, do32 #houveTroca==true

li $t2, 2 #$t2=2

div $t1, $t2

mflo $t1 #nval/2

sll $t1, $t1, 3 #(nval/2)\*8

add $t1, $t0, $t1 #$t1= &(array[nval/2]

l.d $f0, 0($t1) #return array[nval/2]

jr $ra

**AULA 11!**

**1)**

**a)**

void main(void)

{

Double x, res;

Int y;

Print\_str(“Valor de x: “);

X=read\_double();

Print\_str(“Valor de y: “);

Y=read\_int();

Res = xtoy(x, y);

Print\_Str(“Resultado: ”;

Print\_double(res);

}

**b)**

.data

str1: .asciiz "Valor de X: "

str2: .asciiz "Valor de Y: "

str3: .asciiz "Resultado: "

aux: .double 1.0

.text

.globl main

main: la $a0, str1

li $v0, 4

syscall #print\_Str(str1)

li $v0, 7

syscall #$f0=read\_double

la $a0, str2

li $v0, 4

syscall #print\_str(str2)

li $v0, 5

syscall #$v0=read\_int

mov.d $f12, $f0

move $a0, $v0

#

addi $sp, $sp, -4

sw $ra, 0($sp)

jal xtoy

lw $ra, 0($sp)

addi $sp, $sp, 4

#

mov.d $f12, $f0

la $a0, str3

li $v0, 4

syscall #print\_str(str3)

li $v0, 3

syscall #print\_double(res)

jr $ra

xtoy: l.d $f0, aux #result=1.0

li $t0, 0 #i=0

move $t1, $a0 #$t1 = y

mov.d $f2, $f12 #$f2 = x

abs $t2, $t1 #$t2=abs(y)

for: bge $t0, $t2, end\_for

ble $t1, 0, else

mul.d $f0, $f0, $f2

j end\_if

else: div.d $f0, $f0, $f2

end\_if: addi $t0, $t0, 1 #i++

j for

end\_for:

jr $ra

**2)**

**a)**

void main(void)

{

Print\_str(“Valor: “);

Double val, res;

Val = read\_double();

Res= sqrt(val);

Print\_str(“Resultado: “);

Print\_double(val);

}

**b)**

.data

str1: .asciiz "Valor: "

str2: .asciiz "Resultado: "

aux1: .double 0.0

aux2: .double 0.5

aux3: .double 1.0

.text

.globl main

main: la $a0, str1

li $v0, 4

syscall #print\_str(str1)

li $v0, 7

syscall #read\_double()

mov.d $f12, $f0 #sqrt(val)

#

addi $sp, $sp, -4

sw $ra, 0($sp)

jal sqrt

lw $ra, 0($sp)

addi $sp, $sp, 4

#

la $a0, str2

li $v0, 4

syscall #print\_str(str2)

mov.d $f12, $f0

li $v0, 3

syscall #print\_double(val)

jr $ra

sqrt: l.d $f0, aux1

c.le.d $f12, $f0

bc1t end

li $t0, 0 #i=0

l.d $f0, aux3 #xn=1.0

do: mov.d $f2, $f0

div.d $f4, $f12, $f0

add.d $f0, $f0, $f4

l.d $f4, aux2

mul.d $f0, $f0, $f4

c.eq.d $f2, $f0

bc1t end

addi $t0, $t0, 1 #i++

bge $t0, 25, end

j do

end: jr $ra

**3)**

**a)**

stdev: #

addi $sp, $sp, -4

sw $ra, 0($sp)

jal var

lw $ra, 0($sp)

addi $sp, $sp, 4

#

mov.d $f12, $f0

#

addi $sp, $sp, -4

sw $ra, 0($sp)

jal sqrt

lw $ra, 0($sp)

addi $sp, $sp, 4

#

jr $ra

sqrt: l.d $f0, aux1 #if(val<=0.0)

c.le.d $f12, $f0 #return 0.0

bc1t end

li $t0, 0 #i=0

l.d $f0, aux3 #xn=1.0

do: mov.d $f2, $f0

div.d $f4, $f12, $f0 #val/xn

add.d $f0, $f0, $f4 #xn + val/xn

l.d $f4, aux2 #0.5

mul.d $f0, $f0, $f4 #0.5 \* (xn + val/xn)

c.eq.d $f2, $f0 #aux != xn

bc1t end

addi $t0, $t0, 1 #++i

bge $t0, 25, end #i< 25

j do

end: jr $ra

var: move $t0, $a0 #$t0=\*array

move $t1, $a1 #$t1=nval

#

addi $sp, $sp, -12

sw $ra, 0($sp)

sw $t0, 4($sp)

sw $t1, 8($sp)

jal average

lw $ra, 0($sp)

lw $t0, 4($sp)

lw $t1, 8($sp)

addi $sp, $sp, 12

#

move $f4, $f0 #$f4=media

l.d $f6, aux1 #$f6=soma=0.0

li $t2, 0 #i=0

for45: bge $t2, $t1, end\_for45

sll $t3, $t2, 3 #i\*8

add $t3, $t3, $t0 #$t3=&(array[i])

l.d $f12, 0($t3) #$f12=array[i]

sub.d $f12, $f12, $f4 #array[i]-media

li $a0, 2

#

addi $sp, $sp, -32

sw $ra, 0($sp)

sw $t0, 4($sp)

sw $t1, 8($sp)

sw $t2, 12($sp)

s.d $f4, 16($sp)

s.d $f6, 24($sp)

jal xtoy

lw $ra, 0($sp)

lw $t0, 4($sp)

lw $t1, 8($sp)

lw $t2, 12($sp)

l.d $f4, 16($sp)

l.d $f6, 24($sp)

addi $sp, $sp, 32

#

add.d $f6, $f6, $f0 #soma = soma + xtoy(array[i]-media, 2)

j for45

end\_for45:

mtc1 $t1, $f2

cvt.d.w $f2, $f2 #(double) nval

div.d $f0, $f6, $f2 #res=soma/nval

jr $ra

xtoy: l.d $f0, aux3 #result=1.0

li $t0, 0 #i=0

move $t1, $a0 #$t1 = y

mov.d $f2, $f12 #$f2 = x

abs $t2, $t1 #$t2=abs(y)

for: bge $t0, $t2, end\_for

ble $t1, 0, else

mul.d $f0, $f0, $f2

j end\_if

else: div.d $f0, $f0, $f2

end\_if: addi $t0, $t0, 1 #i++

j for

end\_for:

jr $ra

average:

l.d $f4, aux1 #$f4=soma=0.0

move $t0, $a1 #$t0=aux=n

while: ble $t0, 0, end\_while #aux>0

l.d $f6, 0($a0) #$f6=\*array

add.d $f4, $f4, $f6 #soma = soma + \*array

addi $a0, $a0, 8 #array++

addi $t0, $t0, -1 #aux--

j while

end\_while:

mtc1 $a1, $f8

cvt.d.w $f8, $f8 #(double) n

div.d $f0, $f4, $f8 #soma/n

jr $ra

max: l.d $f0, aux1 #max=0

move $t0, $a1 #aux=n

while2: ble $t0, 0, end\_while2

l.d $f6, 0($a0) #$f6=\*array

c.lt.d $f0, $f6

bc1f end\_if

mov.d $f0, $f6 #max = \*array

end\_if: addi $a0, $a0, 8 #array++

addi $t0, $t0, -1 #aux--

j while2

end\_while2:

jr $ra

**b)**

Void main(void)

{

Static double array[11];

Int aux1;

double \*aux2;

double aux3, media, max, mediana, var, stdev;

Unsigned int n=11, i=0;

While(i<11)

{

Print\_str(“Temperatura: “);

Aux1=read\_int();

Aux3 = (double) aux1;

Array[i++] = f2c(aux3);

}

I=0;

While(i<11)

{

Print\_double(array[i++]);

Print\_char(‘\n’);

}

Aux2=array;

Max = max(aux2, n);

Print\_double(max);

Print\_char(‘\n’);

Media=average(aux2, n);

Print\_double(media);

Print\_char(‘\n’);

Mediana=sort(aux2, n);

Print\_double(mediana);

Print\_char(‘\n’);

Var=Var(aux2, n);

Print\_double(var);

Print\_char(‘\n’);

Stdev = stdev(aux2, n);

Print\_double(stdev);

}

**c)**

.data

str1: .asciiz "Temperatura: "

str2: .asciiz "Máximo: "

str3: .asciiz "Média: "

str4: .asciiz "Mediana: "

str5: .asciiz "Variância: "

str6: .asciiz "Desvio Padrão: "

.align 2

array: .space 88

aux1: .double 0.0

aux2: .double 0.5

aux3: .double 1.0

.text

.globl main

main: la $t0, array

li $t1, 11 #n=11

li $t2, 0 #i=0

while1: bge $t2, $t1, end\_while1

la $a0, str1

li $v0, 4

syscall #print\_str(str1)

li $v0, 5

syscall #aux1=read\_int()

mtc1 $v0, $f12

cvt.d.w $f12, $f12 #aux3 =(double) aux1

#

addi $sp, $sp, -16

sw $ra, 0($sp)

sw $t0, 4($sp)

sw $t1, 8($sp)

sw $t2, 12($sp)

jal f2c

lw $ra, 0($sp)

lw $t0, 4($sp)

lw $t1, 8($sp)

lw $t2, 12($sp)

addi $sp, $sp, 16

#

sll $t3, $t2, 3 #i\*8

add $t3, $t0, $t3 #$t3=&(array[i])

s.d $f0, 0($t3) #array[i] = f2c(aux3)

addi $t2, $t2, 1 #i++

j while1

end\_while1:

li $t2, 0 #i=0

while2: bge $t2, $t1, end\_while2

sll $t3, $t2, 3 #i\*8

add $t3, $t0, $t3 #$t3=&(array[i])

l.d $f12, 0($t3) # $f12 = array[i]

li $v0, 3

syscall #print\_double(array[i])

addi $t2, $t2, 1 #i++

li $a0, '\n'

li $v0, 11

syscall #print\_char('\n')

j while2

end\_while2:

la $a0, str2

li $v0, 4

syscall #print\_str(max)

move $a0, $t0

move $a1, $t1

#

addi $sp, $sp, -12

sw $ra, 0($sp)

sw $t0, 4($sp)

sw $t1, 8($sp)

jal max

lw $ra, 0($sp)

lw $t0, 4($sp)

lw $t1, 8($sp)

addi $sp, $sp, 12

#

mov.d $f12, $f0

li $v0, 3

syscall #print\_double(max)

li $a0, '\n'

li $v0, 11

syscall #print\_char('\n')

la $a0, str3

li $v0, 4

syscall #print\_str(media)

move $a0, $t0

move $a1, $t1

#

addi $sp, $sp, -12

sw $ra, 0($sp)

sw $t0, 4($sp)

sw $t1, 8($sp)

jal average

lw $ra, 0($sp)

lw $t0, 4($sp)

lw $t1, 8($sp)

addi $sp, $sp, 12

#

mov.d $f12, $f0

li $v0, 3

syscall #print\_double(media)

li $a0, '\n'

li $v0, 11

syscall #print\_char('\n')

la $a0, str4

li $v0, 4

syscall #print\_str(mediana)

move $a0, $t0

move $a1, $t1

#

addi $sp, $sp, -12

sw $ra, 0($sp)

sw $t0, 4($sp)

sw $t1, 8($sp)

jal sort

lw $ra, 0($sp)

lw $t0, 4($sp)

lw $t1, 8($sp)

addi $sp, $sp, 12

#

mov.d $f12, $f0

li $v0, 3

syscall #print\_double(mediana)

li $a0, '\n'

li $v0, 11

syscall #print\_char('\n')

la $a0, str5

li $v0, 4

syscall #print\_str(var)

move $a0, $t0

move $a1, $t1

#

addi $sp, $sp, -12

sw $ra, 0($sp)

sw $t0, 4($sp)

sw $t1, 8($sp)

jal var

lw $ra, 0($sp)

lw $t0, 4($sp)

lw $t1, 8($sp)

addi $sp, $sp, 12

#

mov.d $f12, $f0

li $v0, 3

syscall #print\_double(var)

li $a0, '\n'

li $v0, 11

syscall #print\_char('\n')

la $a0, str6

li $v0, 4

syscall #print\_str(stdev)

move $a0, $t0

move $a1, $t1

#

addi $sp, $sp, -4

sw $ra, 0($sp)

jal stdev

lw $ra, 0($sp)

addi $sp, $sp, 4

#

mov.d $f12, $f0

li $v0, 3

syscall #print\_double(stdev)

jr $ra

f2c:

li $t0, 5

mtc1 $t0, $f2

cvt.d.w $f2, $f2 #(double) 5

li $t0, 9

mtc1 $t0, $f4

cvt.d.w $f4, $f4 #(double) 9

li $t0, 32

mtc1 $t0, $f6

cvt.d.w $f6, $f6 #(double) 32

div.d $f8, $f2, $f4 #$f8 =5/9

sub.d $f10, $f12, $f6 #$f10 = ft-32

mul.d $f0, $f8, $f10 #5/9 \* (ft-32)

jr $ra

stdev: #

addi $sp, $sp, -4

sw $ra, 0($sp)

jal var

lw $ra, 0($sp)

addi $sp, $sp, 4

#

mov.d $f12, $f0

#

addi $sp, $sp, -4

sw $ra, 0($sp)

jal sqrt

lw $ra, 0($sp)

addi $sp, $sp, 4

#

jr $ra

sqrt: l.d $f0, aux1 #if(val<=0.0)

c.le.d $f12, $f0 #return 0.0

bc1t end1

li $t0, 0 #i=0

l.d $f0, aux3 #xn=1.0

do1: mov.d $f2, $f0

div.d $f4, $f12, $f0 #val/xn

add.d $f0, $f0, $f4 #xn + val/xn

l.d $f4, aux2 #0.5

mul.d $f0, $f0, $f4 #0.5 \* (xn + val/xn)

c.eq.d $f2, $f0 #aux != xn

bc1t end1

addi $t0, $t0, 1 #++i

bge $t0, 25, end1 #i< 25

j do1

end1: jr $ra

var: move $t0, $a0 #$t0=\*array

move $t1, $a1 #$t1=nval

#

addi $sp, $sp, -12

sw $ra, 0($sp)

sw $t0, 4($sp)

sw $t1, 8($sp)

jal average

lw $ra, 0($sp)

lw $t0, 4($sp)

lw $t1, 8($sp)

addi $sp, $sp, 12

#

mov.d $f4, $f0 #$f4=media

l.d $f6, aux1 #$f6=soma=0.0

li $t2, 0 #i=0

for1: bge $t2, $t1, end\_for1

sll $t3, $t2, 3 #i\*8

add $t3, $t3, $t0 #$t3=&(array[i])

l.d $f12, 0($t3) #$f12=array[i]

sub.d $f12, $f12, $f4 #array[i]-media

li $a0, 2

#

addi $sp, $sp, -32

sw $ra, 0($sp)

sw $t0, 4($sp)

sw $t1, 8($sp)

sw $t2, 12($sp)

s.d $f4, 16($sp)

s.d $f6, 24($sp)

jal xtoy

lw $ra, 0($sp)

lw $t0, 4($sp)

lw $t1, 8($sp)

lw $t2, 12($sp)

l.d $f4, 16($sp)

l.d $f6, 24($sp)

addi $sp, $sp, 32

#

add.d $f6, $f6, $f0 #soma = soma + xtoy(array[i]-media, 2)

addi $t2, $t2, 1 #i++

j for1

end\_for1:

mtc1 $t1, $f2

cvt.d.w $f2, $f2 #(double) nval

div.d $f0, $f6, $f2 #res=soma/nval

jr $ra

xtoy: l.d $f0, aux3 #result=1.0

li $t0, 0 #i=0

move $t1, $a0 #$t1 = y

mov.d $f2, $f12 #$f2 = x

abs $t2, $t1 #$t2=abs(y)

for2: bge $t0, $t2, end\_for2

ble $t1, 0, else1

mul.d $f0, $f0, $f2

j end\_if1

else1: div.d $f0, $f0, $f2

end\_if1: addi $t0, $t0, 1 #i++

j for2

end\_for2:

jr $ra

average:

l.d $f4, aux1 #$f4=soma=0.0

move $t0, $a1 #$t0=aux=n

while3: ble $t0, 0, end\_while3 #aux>0

l.d $f6, 0($a0) #$f6=\*array

add.d $f4, $f4, $f6 #soma = soma + \*array

addi $a0, $a0, 8 #array++

addi $t0, $t0, -1 #aux--

j while3

end\_while3:

mtc1 $a1, $f8

cvt.d.w $f8, $f8 #(double) n

div.d $f0, $f4, $f8 #soma/n

jr $ra

max: l.d $f0, aux1 #max=0

move $t0, $a1 #aux=n

while4: ble $t0, 0, end\_while4

l.d $f6, 0($a0) #$f6=\*array

c.lt.d $f0, $f6

bc1f end\_if2

mov.d $f0, $f6 #max = \*array

end\_if2: addi $a0, $a0, 8 #array++

addi $t0, $t0, -1 #aux--

j while4

end\_while4:

jr $ra

sort:

move $t0, $a0 #$t0=\*array

move $t1, $a1 #$t1=nval

do2: li $t2, 0 #houveTroca=false(0)

li $t3, 0 #i=0

addi $t4, $t1, -1 #nval-1

for3: bge $t3, $t4, end\_for3

sll $t5, $t3, 3 #i\*8

add $t5, $t0, $t5 #$t5= &(array[i])

l.d $f0, 0($t5) #$f0=array[i]

add $t6, $t3, 1 #i+1

sll $t6, $t6, 3 #(i+1)\*8

add $t6, $t0, $t6 #$t5= &(array[i+1])

l.d $f2, 0($t6) #$f2=array[i+1]

c.le.d $f0, $f2

bc1t end\_if3 #array[i] > array[i+1]

mov.d $f4, $f0 #aux=array[i]

s.d $f2, 0($t5) #array[i] = array[i+1]

s.d $f4, 0($t6) #array[i+1] = aux(array[i])

li $t2, 1 #houveTrocas=1

end\_if3:

addi $t3, $t3, 1 #i++

j for3

end\_for3:

beq $t2, 1, do2 #houveTroca==true

li $t2, 2 #$t2=2

div $t1, $t2

mflo $t1 #nval/2

sll $t1, $t1, 3 #(nval/2)\*8

add $t1, $t0, $t1 #$t1= &(array[nval/2]

l.d $f0, 0($t1) #return array[nval/2]

jr $ra

**AULA 12!**

**1)**

**a)**

.data

nmec: .word 72343

nome: .asciiz "Maria"

nome2: .asciiz "Simplesmente"

grade: .float 17.9

stg: .space 44

aux1: .space 18

aux2: .space 15

str1: .asciiz "\nN. Mec: "

str2: .asciiz "\nNome: "

str3: .asciiz "\nNota: "

.align 2

.text

.globl main

main: la $t0, stg #$t0=&(stg(id\_number)

la $t1, nmec

lw $t1, 0($t1) #$t1=72343

sw $t1, 0($t0) #$t1=stg(id\_number)72343

addi $t0, $t0, 4 #$t0=&(stg(first\_name))

la $t1, nome #$t1=nome

li $t3, 0 #i=0

while1: lb $t2, 0($t1) #$t2 = \*nome

sb $t2, 0($t0) #stg(\*first\_name=\*nome)

addi $t1, $t1, 1 #nome++

addi $t0, $t0, 1 #stg(first\_name++)

addi $t3, $t3, 1 #i++

beq $t2, '\0', end\_while1

j while1

end\_while1:

li $t4, 18

sub $t3, $t4, $t3 #18-i

while2: beq $t3, 0, end\_while2

addi $t0, $t0, 1 #stg(first\_name++)

addi $t3, $t3, -1 #i--

j while2

end\_while2:

la $t1, nome2 #$t1=nome2

li $t3, 0 #i=0

while3: lb $t2, 0($t1) #$t2=\*nome2

sb $t2, 0($t0) #\*stg(first\_name)=\*nome2

addi $t0, $t0, 1 #stg(last\_name++)

addi $t1, $t1, 1 #nome2++

addi $t3, $t3, 1 #i++

beq $t2, '\0', end\_while3

j while3

end\_while3:

sub $t3, $t4, $t3 #18-i

while4: beq $t3, 0, end\_while4

addi $t0, $t0, 1 #stg++

addi $t3, $t3, -1 #i--

j while4

end\_while4:

la $t0, stg #$t0=&(stg)

addi $t0, $t0, 40 #$t0=&(stg(grade))

l.d $f0, grade #$f0=17.9

s.s $f0, 0($t0) #stg(grade)=17.9

############################################

la $t0, stg #$t0=&(stg(id\_number)

la $a0, str1

li $v0, 4

syscall#print\_str(str1)

lw $a0, 0($t0) #$t1=stg(id\_number)72343

li $v0, 36

syscall #print\_intu10(id\_number)

###

addi $t0, $t0, 22 #$t0=&(stg(last\_name))

la $t1, aux1 #$t1=&(aux1)

while5: lb $t2, 0($t0) #$t2 = \*aux1

sb $t2, 0($t1) #\*nome =\*stg(last\_name)

addi $t1, $t1, 1 #nome++

addi $t0, $t0, 1 #stg(last\_name++)

beq $t2, '\0', end\_while5

j while5

end\_while5:

la $a0, str2

li $v0, 4

syscall #print\_str(str2)

la $a0, aux1

li $v0, 4

syscall #print\_str(stg.last\_name)

###

la $t0, stg #$t0=&(stg)

addi $t0, $t0, 4 #$t0 = &(stg(first\_name))

la $t1, aux2 #$t1 =&(aux2)

while6: lb $t2, 0($t0)

sb $t2, 0($t1) #\*nome2=\*stg(first\_name)

addi $t0, $t0, 1 #stg(first\_name++)

addi $t1, $t1, 1 #nome2++

beq $t2, '\0', end\_while6

j while6

end\_while6:

li $a0, ','

li $v0, 11

syscall #print\_char(',')

la $a0, aux2

li $v0, 4

syscall #print\_str(stg.first\_name)

###

la $t0, stg #$t0=&(stg)

addi $t0, $t0, 40 #$t0 = &(stg(grade))

la $a0, str3

li $v0, 4

syscall #print\_str(str3)

l.s $f12, 0($t0) #$f12=stg(grade)=17.9

li $v0, 2

syscall #print\_float(stg.grade)

jr $ra

**b)**

typedef struct

{

unsigned int id\_number;

char first\_name[18];

char last\_name[15]

float grade;

} student;

int main(void)

{

// define a estrutura "stg" no segmento de dados

static student stg = {);

print\_str("N. Mec: ");

stg.id\_number = read\_int();

print\_str("Primeiro Nome: ");

read\_str(stg.first\_name, 18);

print\_str("Último Nome: ");

read\_str(stg.last\_name, 18);

print\_str(“Nota: “);

stg.grade = read\_float();

print\_str("\nN. Mec: ");

print\_intu10(stg.id\_number);

print\_str("\nNome: ");

print\_str(stg.last\_name);

print\_char(',');

print\_str(stg.first\_name);

print\_str("\nNota: ");

print\_float(stg.grade);

return 0;

}

**c)**

.data

stg: .space 44

aux1: .space 18

aux2: .space 15

str1: .asciiz "N. Mec: "

str2: .asciiz "Primeiro Nome: "

str3: .asciiz "Último Nome: "

str4: .asciiz "Nota: "

str5: .asciiz "N. Mec: "

str6: .asciiz "\nNome: "

str7: .asciiz "Nota: "

.align 2

.text

.globl main

main: la $a0, str1

li $v0, 4

syscall #print\_str(str1)

la $t0, stg #$t0=&(stg(id\_number)

la $v0, 5

syscall #read\_int()

sw $v0, 0($t0) #stg(id\_number)=read\_int()

la $a0, str2

li $v0, 4

syscall #print\_str(str2)

la $a0, aux1

li $a1, 18

li $v0, 8

syscall #read\_str()

addi $t0, $t0, 4 #$t0=&(stg(first\_name))

la $t1, aux1 #$t1=nome

while1: lb $t2, 0($t1) #$t2 = \*nome

sb $t2, 0($t0) #stg(\*first\_name=\*nome)

addi $t1, $t1, 1 #nome++

addi $t0, $t0, 1 #stg(first\_name++)

beq $t2, '\0', end\_while1

j while1

end\_while1:

la $a0, str3

li $v0, 4

syscall #print\_str(str3)

la $a0, aux2

li $a1, 15

li $v0, 8

syscall #read\_str()

la $t0, stg #$t0=&(stg)

addi $t0, $t0, 22 #$t0=&(stg.last\_name)

la $t1, aux2 #$t1=nome2

while3: lb $t2, 0($t1) #$t2=\*nome2

sb $t2, 0($t0) #\*stg(last\_name)=\*nome2

addi $t0, $t0, 1 #stg(last\_name++)

addi $t1, $t1, 1 #nome2++

beq $t2, '\0', end\_while3

j while3

end\_while3:

la $a0, str4

li $v0, 4

syscall #print\_str(str4)

li $v0, 6

syscall #read\_float()

la $t0, stg #$t0=&(stg)

addi $t0, $t0, 40 #$t0=&(stg(grade))

s.s $f0, 0($t0) #stg(grade)=17.9

############################################

la $t0, stg #$t0=&(stg(id\_number)

la $a0, str5

li $v0, 4

syscall#print\_str(str5)

lw $a0, 0($t0) #$t1=stg(id\_number)72343

li $v0, 36

syscall #print\_intu10(id\_number)

###

addi $t0, $t0, 22 #$t0=&(stg(last\_name))

la $t1, aux1 #$t1=&(aux1)

while5: lb $t2, 0($t0) #$t2 = \*aux1

sb $t2, 0($t1) #\*nome =\*stg(last\_name)

addi $t1, $t1, 1 #nome++

addi $t0, $t0, 1 #stg(last\_name++)

beq $t2, '\0', end\_while5

j while5

end\_while5:

la $a0, str6

li $v0, 4

syscall #print\_str(str6)

la $a0, aux1

li $v0, 4

syscall #print\_str(stg.last\_name)

###

la $t0, stg #$t0=&(stg)

addi $t0, $t0, 4 #$t0 = &(stg(first\_name))

la $t1, aux2 #$t1 =&(aux2)

while6: lb $t2, 0($t0)

sb $t2, 0($t1) #\*nome2=\*stg(first\_name)

addi $t0, $t0, 1 #stg(first\_name++)

addi $t1, $t1, 1 #nome2++

beq $t2, '\0', end\_while6

j while6

end\_while6:

li $a0, ','

li $v0, 11

syscall #print\_char(',')

la $a0, aux2

li $v0, 4

syscall #print\_str(stg.first\_name)

###

la $t0, stg #$t0=&(stg)

addi $t0, $t0, 40 #$t0 = &(stg(grade))

la $a0, str7

li $v0, 4

syscall #print\_str(str7)

l.s $f12, 0($t0) #$f12=stg(grade)=17.9

li $v0, 2

syscall #print\_float(stg.grade)

jr $ra

**2)**

**a)**

.data

sta: .space 176

stu: .space 44

str1: .asciiz "\nMedia: "

.align 2

.text

.globl main

main: la $a0, sta #st\_arrar

li $a1, 4 #max\_students

#

addi $sp, $sp, -4

sw $ra, 0($sp)

jal read\_data

lw $ra, 0($sp)

addi $sp, $sp, 4

#

la $a0, sta #st\_array

li $a1, 4 #max\_students

#$f12 = $media

#

addi $sp, $sp, -4

sw $ra, 0($sp)

jal max

lw $ra, 0($sp)

addi $sp, $sp, 4

#

move $t0, $v0 #$t0=pmax=max(st\_array, max\_students, $f12=&media)

la $a0, str1

li $v0, 4

syscall #print\_str(str1)

#$f12 = media

li $v0, 6

syscall #print\_float(media)

move $a0, $t0 #pmax

#

addi $sp, $sp, -4

sw $ra, 0($sp)

jal print\_student

lw $ra, 0($sp)

addi $sp, $sp, 4

#

jr $ra

**b)**

read\_data:

move $t0, $a0 #$t0 = &(st)

move $t1, $a1 #$t1 = ns

li $t2, 0 #i=0

for1: bge $t2, $t1, end\_for1

li $v0, 4

syscall #print\_str(str1)

la $t0, stg #$t0=&(st[i].id\_number)

la $v0, 5

syscall #read\_int()

mulo $t3, $t2, 44 #$t3=i\*44

add $t3, $t0, $t3 #$t3 = st[i].id\_number

sw $v0, 0($t3) #st[i].id\_number=read\_int()

la $a0, str2

li $v0, 4

syscall #print\_str(str2)

la $a0, aux1

li $a1, 18

li $v0, 8

syscall #read\_str()

addi $t3, $t3, 4 #$t3=st[i].first\_name

la $t4, aux1 #$t4=nome

while1: lb $t5, 0($t4) #$t5 = \*nome

sb $t5, 0($t3) #st[i].\*first\_name=\*nome

addi $t4, $t4, 1 #nome++

addi $t3, $t3, 1 #st[i].first\_name++

beq $t5, '\0', end\_while1

j while1

end\_while1:

la $a0, str3

li $v0, 4

syscall #print\_str(str3)

la $a0, aux2

li $a1, 15

li $v0, 8

syscall #read\_str()

mulo $t3, $t2, 44 #i\*44

addi $t3, $t3, 22 #$t3=&(st[i].last\_name)

la $t4, aux2 #$t5=nome2

while3: lb $t5, 0($t4) #$t5=\*nome2

sb $t5, 0($t3) #\*st[i].last\_name=\*nome2

addi $t3, $t3, 1 #st[i].last\_name++

addi $t4, $t4, 1 #nome2++

beq $t5, '\0', end\_while3

j while3

end\_while3:

la $a0, str4

li $v0, 4

syscall #print\_str(str4)

li $v0, 6

syscall #read\_float()

mul $t3, $t2, 44 #i\*44

addi $t3, $t3, 40 #$t3=&(st[i].grade)

s.s $f0, 0($t3) #st[i].grade=$f0

jr $ra

**c)**

max:

move $t0, $a0 #$t0=\*st

move $t1, $a1 #$t1=ns

mov.s $f0, $f12 #$f0=&(\*media)

l.s $f2, maxg #$f2=max\_grade=-20.0

mtc1 $0, $f4, #$f4=sum=0.0

la $t2, stu #$t2=\*pmax

move $t3, $t0 #p=st

mulo $t4, $t1, 44 #ns\*44

add $t4, $t0, $t4 #st+ns

for2: bge $t3, $t4, end\_for2

addi $t5, $t3, 40 #$t5=&(p->grade)

l.s $f6, 0($t5) #$f4=p->grade

add.s $f4, $f4, $f6 #sum += p->grade

c.le.s $f6, $f2 #p->grrade > max\_grade

bc1t end\_if1

mov.s $f2, $f6

li $t5, 0 #i=0

move $t6, $t3 #aux1=p

move $t7, $t2 #aux2=pmax

whilecopy:

bge $t5, 44, end\_whilecopy

lb $t8, 0($t6) #$t7=p

sb $t8, 0($t7) #pmax=p

addi $t5, $t5, 1 #i++

addi $t6, $t6, 1 #aux1++

addi $t7, $t7, 1 #aux2++

j whilecopy

end\_whilecopy:

addi $t3, $t3, 44 #p++

j for2

end\_for2:

mtc1 $t1, $f2

cvt.s.w $f2, $f2 #(float)ns

div.s $f0, $f4, $f2 #\*media = sum / (float)ns

move $v0, $t2

jr $ra

**d)**

print\_student:

move $t3, $a0 #$t3=&(p->id\_number)

move $t0, $a0 #$t0=&(p->id\_number)

la $a0, str5

li $v0, 4

syscall#print\_str(str5)

lw $a0, 0($t0) #$t1=p->id\_number

li $v0, 36

syscall #print\_intu10(p->id\_number)

###

addi $t0, $t0, 4 #$t0=&(p->first\_name))

la $t1, nome1 #$t1=&(nome1)

while5: lb $t2, 0($t0) #$t2 = \*aux1

sb $t2, 0($t1) #\*nome =\*p->first\_name)

addi $t1, $t1, 1 #nome++

addi $t0, $t0, 1 #p->first\_name++

beq $t2, '\0', end\_while5

j while5

end\_while5:

la $a0, str6

li $v0, 4

syscall #print\_str(str6)

la $a0, nome1

li $v0, 4

syscall #print\_str(p->first\_name)

###

move $t0, $t3 #$t0=&(\*p)

addi $t0, $t0, 22 #$t0 = &(p->last\_name))

la $t1, nome2 #$t1 =&(nome2)

while6: lb $t2, 0($t0)

sb $t2, 0($t1) #\*nome2=\*p->last\_name)

addi $t0, $t0, 1 #p->last\_name++)

addi $t1, $t1, 1 #nome2++

beq $t2, '\0', end\_while6

j while6

end\_while6:

la $a0, nome2

li $v0, 4

syscall #print\_str(p->last\_name)

###

move $t0, $t3 #$t0=&(\*p)

addi $t0, $t0, 40 #$t0 = &(p->grade))

la $a0, str7

li $v0, 4

syscall #print\_str(str7)

l.s $f12, 0($t0) #$f12=p->grade

li $v0, 2

syscall #print\_float(p->grade)

jr $ra

**e)**

.data

sta: .space 176

stu: .space 44

nome1: .space 18

nome2: .space 15

str0: .asciiz "\nMedia: "

str1: .asciiz "N. Mec: "

str2: .asciiz "Primeiro Nome: "

str3: .asciiz "Último Nome: "

str4: .asciiz "Nota: "

str5: .asciiz "\nN. Mec: "

str6: .asciiz "\nNome: "

str7: .asciiz "Nota: "

maxg: .float -20.0

.align 2

.text

.globl main

main: la $a0, sta #st\_arrar

li $a1, 4 #max\_students

#

addi $sp, $sp, -4

sw $ra, 0($sp)

jal read\_data

lw $ra, 0($sp)

addi $sp, $sp, 4

#

la $a0, sta #st\_array

li $a1, 4 #max\_students

#$f12 = $media

#

addi $sp, $sp, -4

sw $ra, 0($sp)

jal max

lw $ra, 0($sp)

addi $sp, $sp, 4

#

move $t0, $v0 #$t0=pmax=max(st\_array, max\_students, $f12=&media)

la $a0, str0

li $v0, 4

syscall #print\_str(str0)

#$f12 = media

li $v0, 2

syscall #print\_float(media)

move $a0, $t0 #pmax

#

addi $sp, $sp, -4

sw $ra, 0($sp)

jal print\_student

lw $ra, 0($sp)

addi $sp, $sp, 4

#

jr $ra

################

print\_student:

move $t3, $a0 #$t3=&(p->id\_number)

move $t0, $a0 #$t0=&(p->id\_number)

la $a0, str5

li $v0, 4

syscall#print\_str(str5)

lw $a0, 0($t0) #$t1=p->id\_number

li $v0, 36

syscall #print\_intu10(p->id\_number)

###

addi $t0, $t0, 4 #$t0=&(p->first\_name))

la $t1, nome1 #$t1=&(nome1)

while5: lb $t2, 0($t0) #$t2 = \*aux1

sb $t2, 0($t1) #\*nome =\*p->first\_name)

addi $t1, $t1, 1 #nome++

addi $t0, $t0, 1 #p->first\_name++

beq $t2, '\0', end\_while5

j while5

end\_while5:

la $a0, str6

li $v0, 4

syscall #print\_str(str6)

la $a0, nome1

li $v0, 4

syscall #print\_str(p->first\_name)

###

move $t0, $t3 #$t0=&(\*p)

addi $t0, $t0, 22 #$t0 = &(p->last\_name))

la $t1, nome2 #$t1 =&(nome2)

while6: lb $t2, 0($t0)

sb $t2, 0($t1) #\*nome2=\*p->last\_name)

addi $t0, $t0, 1 #p->last\_name++)

addi $t1, $t1, 1 #nome2++

beq $t2, '\0', end\_while6

j while6

end\_while6:

la $a0, nome2

li $v0, 4

syscall #print\_str(p->last\_name)

###

move $t0, $t3 #$t0=&(\*p)

addi $t0, $t0, 40 #$t0 = &(p->grade))

la $a0, str7

li $v0, 4

syscall #print\_str(str7)

l.s $f12, 0($t0) #$f12=p->grade

li $v0, 2

syscall #print\_float(p->grade)

jr $ra

##################

max:

move $t0, $a0 #$t0=\*st

move $t1, $a1 #$t1=ns

mov.s $f0, $f12 #$f0=&(\*media)

l.s $f2, maxg #$f2=max\_grade=-20.0

mtc1 $0, $f4, #$f4=sum=0.0

la $t2, stu #$t2=\*pmax

move $t3, $t0 #p=st

mulo $t4, $t1, 44 #ns\*44

add $t4, $t0, $t4 #st+ns

for2: bge $t3, $t4, end\_for2

addi $t5, $t3, 40 #$t5=&(p->grade)

l.s $f6, 0($t5) #$f4=p->grade

add.s $f4, $f4, $f6 #sum += p->grade

c.le.s $f6, $f2 #p->grrade > max\_grade

bc1t end\_if1

mov.s $f2, $f6

li $t5, 0 #i=0

move $t6, $t3 #aux1=p

move $t7, $t2 #aux2=pmax

whilecopy:

bge $t5, 44, end\_whilecopy

lb $t8, 0($t6) #$t7=p

sb $t8, 0($t7) #pmax=p

addi $t5, $t5, 1 #i++

addi $t6, $t6, 1 #aux1++

addi $t7, $t7, 1 #aux2++

j whilecopy

end\_whilecopy:

end\_if1:

addi $t3, $t3, 44 #p++

j for2

end\_for2:

mtc1 $t1, $f2

cvt.s.w $f2, $f2 #(float)ns

div.s $f0, $f4, $f2 #\*media = sum / (float)ns

move $v0, $t2

jr $ra

###################

read\_data:

move $t0, $a0 #$t0 = &(st)

move $t1, $a1 #$t1 = ns

li $t2, 0 #i=0

for1: bge $t2, $t1, end\_for1

la $a0, str1

li $v0, 4

syscall #print\_str(str1)

la $v0, 5

syscall #read\_int()

mulo $t3, $t2, 44 #$t3=i\*44

add $t3, $t0, $t3 #$t3 = st[i].id\_number

sw $v0, 0($t3) #st[i].id\_number=read\_int()

la $a0, str2

li $v0, 4

syscall #print\_str(str2)

la $a0, nome1

li $a1, 18

li $v0, 8

syscall #read\_str()

addi $t3, $t3, 4 #$t3=&(st[i].first\_name)

la $t4, nome1 #$t4=nome1

while1: lb $t5, 0($t4) #$t5 = \*nome1

sb $t5, 0($t3) #st[i].\*first\_name=\*nome

addi $t4, $t4, 1 #nome++

addi $t3, $t3, 1 #st[i].first\_name++

beq $t5, '\0', end\_while1

j while1

end\_while1:

la $a0, str3

li $v0, 4

syscall #print\_str(str3)

la $a0, nome2

li $a1, 15

li $v0, 8

syscall #read\_str()

mulo $t3, $t2, 44 #i\*44

addi $t3, $t3, 22 #i\*44+22

add $t3, $t0, $t3 #$t3=&(st[i].last\_name)

la $t4, nome2 #$t4=nome2

while3: lb $t5, 0($t4) #$t5=\*nome2

sb $t5, 0($t3) #\*st[i].last\_name=\*nome2

addi $t3, $t3, 1 #st[i].last\_name++

addi $t4, $t4, 1 #nome2++

beq $t5, '\0', end\_while3

j while3

end\_while3:

la $a0, str4

li $v0, 4

syscall #print\_str(str4)

li $v0, 6

syscall #read\_float()

mul $t3, $t2, 44 #i\*44

addi $t3, $t3, 40 #$t3=&(st[i].grade)

s.s $f0, 0($t3) #st[i].grade=$f0

addi $t2, $t2, 1 #i++

j for1

end\_for1:

jr $ra