Primeiro Trabalho Prático de AOC I
Assembly MIPS

Pablo Cecilio Oliveira
Marcilene Reis

1 Introdução

Mambojambo introduzindo sobre o que é o documento, as questões, Mars, etc.

Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Ut purus elit, vestibulum ut, placerat ac, adipiscing vitae, felis. Curabitur dictum gravida mauris. Nam arcu libero, nonummy eget, consectetuer id, vulputate a, magna. Donec vehicula augue eu neque. Pellentesque habitant morbi tristique senectus et netus et malesuada fames ac turpis egestas. Mauris ut leo. Cras viverra metus rhoncus sem. Nulla et lectus vestibulum urna fringilla ultrices. Phasellus eu tellus sit amet tortor gravida placerat. Integer sapien est, iaculis in, pretium quis, viverra ac, nunc. Praesent eget sem vel leo ultrices bibendum. Aenean faucibus. Morbi dolor nulla, malesuada eu, pulvinar at, mollis ac, nulla. Curabitur auctor semper nulla. Donec varius orci eget risus. Duis nibh mi, congue eu, accumsan eleifend, sagittis quis, diam. Duis eget orci sit amet orci dignissim rutrum.

2 Implementação

2.1 Questão 1

O procedimento para calcular o *cosseno* de um ângulo segundo a série de Taylor abaixo:

$$\cos x = \sum_{n=0}^{\infty} \frac{(-1)^n}{(2n)!} x^{2n}$$

A função *seno* chama funções para calcular o fatorial e a função potência de x. O usuário digita o ângulo em radianos e a quantidade de termos da série ($n > 0$).

2.2 Questão 2

O programa lê uma `string` de um arquivo denominado `entrada.txt` e gera um novo arquivo de nome `saida.txt` contendo uma nova `string` processada.

A `string` gerada no arquivo de saída possui seus caracteres maiúsculos trocados por minúsculos, e seus caracteres minúsculos trocados por maiúsculos.

Para iniciar o processamento, inicialmente os dados dos arquivos a serem carregados são declarados, assim como é definido o espaço reservado para o buffer de leitura do arquivo de entrada.

```
.data
fin:     .asciiz "entrada.txt"
fout:    .asciiz "saida.txt"
string:  .space 1024
```

Inicia .text. A chamada para os procedimentos e a passagem das referencias.

Escopo principal do programa

`main:`	
`la $a0,fin`	atribui o arquivo de entrada a `$a0`
`jal leArquivo`	inicia o procedimento
`la $a0,string`	atribui a string a `$a0`
`jal manipulaString`	inicia o procedimento
`la $a0,fout`	atribui o arquivo de saida a `$a0`
`jal salvaArquivo`	inicia o procedimento
`li $v0,10`	system call: Saia do programa
`syscall`	Saia!

Procedimento `leArquivo` mais comentários a respeito de seu funcionamento por etapas e breve sumario de conclusão desse.

Procedimento para ler o arquivo

`leArquivo:`	
`li $v0,13`	system call: Abre o arquivo
`li $a1,0`	flag para somente leitura
`li $a2,0`	modo ignorado
`syscall`	Abra o arquivo!
`move $t0,$v0`	Salva o descriptor em `$t0`
`li $v0,14`	system call: Lendo o arquivo
`move $a0,$t0`	Carrega o descriptor do arquivo
`la $a1,string`	endereco do buffer
`li $a2,255`	hardcoded buffer length
`syscall`	Leia o arquivo!
`move $s0,$v0`	Salva strlen em `$s0`
`li $v0,16`	system call: Fecha o arquivo
`syscall`	Feche o arquivo!
`jr $ra`	Retorna do procediemnto

Procedimento `manipulaString` mais comentários a respeito de seu funcionamento por etapas e breve sumario de conclusão desse.

Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Ut purus elit, vestibulum ut, placerat ac, adipiscing vitae, felis. Curabitur dictum gravida mauris. Nam arcu libero, nonummy eget, consectetuer id, vulputate a, magna. Donec vehicula augue eu neque. Pellentesque habitant morbi tristique senectus et netus et malesuada fames ac turpis egestas. Mauris ut leo. Cras viverra metus rhoncus sem. Nulla et lectus vestibulum urna

fringilla ultrices. Phasellus eu tellus sit amet tortor gravida placerat. Integer sapien est, iaculis in, pretium quis, viverra ac, nunc. Praesent eget sem vel leo ultrices bibendum. Aenean faucibus. Morbi dolor nulla, malesuada eu, pulvinar at, mollis ac, nulla. Curabitur auctor semper nulla. Donec varius orci eget risus. Duis nibh mi, congue eu, accumsan eleifend, sagittis quis, diam. Duis eget orci sit amet orci dignissim rutrum.

Tabela 2.2.1: Tabela ASCII para A-Z e a-z

65	A	69	E	73	I	77	M	81	Q	85	U	89	Y
66	B	70	F	74	J	78	N	82	R	86	V	90	Z
67	C	71	G	75	K	79	O	83	S	87	W		
68	D	72	H	76	L	80	P	84	T	88	X		
97	a	101	e	105	i	109	m	113	q	117	u	121	y
98	b	102	f	106	j	110	n	114	r	118	v	122	z
99	c	103	g	107	k	111	o	115	s	119	w		
100	d	104	h	108	l	112	p	116	t	120	x		

Fonte: Wikipedia contributors [2]

Procedimento para alterar a string

`manipulaString:`	
`Loop:`	loop para cada caractere da string
`lb $t2,($a0)`	Aponta $t2 para a posição de endereço de `$a0`
`beq $t2,$zero,End`	Se $t2 conter NULL, a string terminou, `End`
`slti $t1,$t2,90`	$t2 e menor que 90(Z)?
`bne $t1,$zero,Upper`	Se $t1 for 1 então 1 != 0, vai pra `Upper`
`slti $t1,$t2,122`	$t2 e menor que 122(z)?
`bne $t1,$zero,Lower`	Se $t1 for 1 então 1 != 0, vai pra `Lower`
`j Next`	Va pra Next
`Upper:`	rotina para uppercase
`slti $t1,$t2,65`	$t2 e menor que 65(A)?
`bne $t1,$zero,Next`	Se $t1 for 1 então não é uma letra, `Next`
`addi $t2,$t2,32`	Soma 32 a $t2
`sb $t2,($a0)`	armazena o valor na posição em `$a0`
`j Next`	Va pra Next
`Lower:`	rotina para lowercase
`slti $t1,$t2,97`	$t2 e menor que 97(a)?
`bne $t1,$zero,Next`	Se $t1 for 1 então não é uma letra, Next
`addi $t2,$t2,-32`	Soma -32 a `$t2`
`sb $t2,($a0)`	armazena o valor na posição em `$a0`
`Next:`	Continua a interação
`addi $a0,$a0,1`	Incrementa o endereço de `$a0`
`j Loop`	Va para Loop
`End:`	Termino do Loop
`jr $ra`	Retorna do procedimento

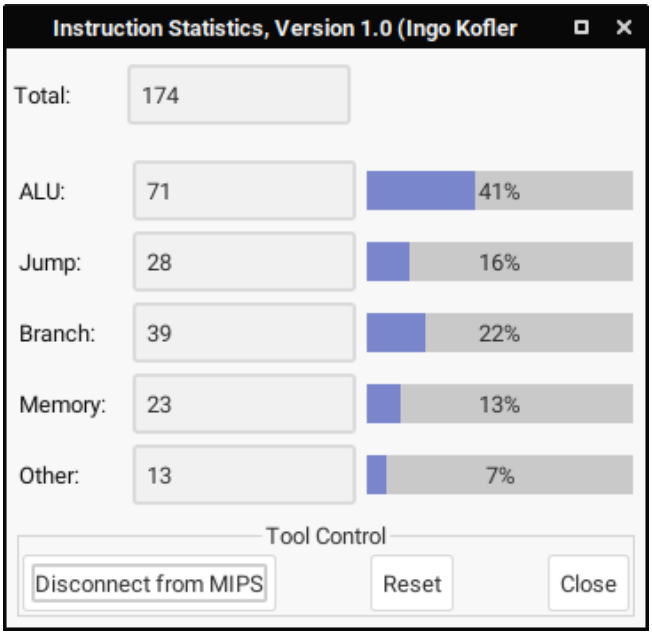
Procedimento `salvaArquivo` mais comentários a respeito de seu funcionamento por etapas e breve sumario de conclusão desse.

Procedimento para salvar o arquivo

`salvaArquivo:`	
`li $v0, 13`	system call: Abre o arquivo
`li $a1, 1`	flag para escrita
`li $a2, 0`	modo ignorado
`syscall`	Abra o arquivo!
`move $t0,$v0`	Salva o descriptor em $t0
`li $v0,15`	system call: Escrevendo no arquivo
`move $a0,$t0`	Carrega o descriptor do arquivo
`la $a1,string`	endereço do buffer
`move $a2,$s0`	carrega strlen de $s0
`syscall`	Escreva no arquivo!
`li $v0,16`	system call: Fecha o arquivo
`syscall`	Feche o arquivo!
`jr $ra`	Retorna do procedimento

Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Ut purus elit, vestibulum ut, placerat ac, adipiscing vitae, felis. Curabitur dictum gravida mauris. Nam arcu libero, nonummy eget, consectetuer id, vulputate a, magna. Donec vehicula augue eu neque. Pellentesque habitant morbi tristique senectus et netus et malesuada fames ac turpis egestas. Mauris ut leo. Cras viverra metus rhoncus sem. Nulla et lectus vestibulum urna fringilla ultrices. Phasellus eu tellus sit amet tortor gravida placerat. Integer sapien est, iaculis in, pretium quis, viverra ac, nunc. Praesent eget sem vel leo ultrices bibendum. Aenean faucibus. Morbi dolor nulla, malesuada eu, pulvinar at, mollis ac, nulla. Curabitur auctor semper nulla. Donec varius orci eget risus. Duis nibh mi, congue eu, accumsan eleifend, sagittis quis, diam. Duis eget orci sit amet orci dignissim rutrum.

Figura 2.2.1: Questão 2, estatísticas das instruções.



Fonte: Mars Instruction Statistic Tool

Referências

[1] Missouri State University. Syscall functions available in mars. http://courses.missouristate.edu/kenvollmar/mars/help/syscallhelp.html. [Acesso em: 13-Outubro-2018].

[2] Wikipedia contributors. Ascii — Wikipedia, the free encyclopedia. https://en.wikipedia.org/w/index.php?title=ASCII&oldid=863023280, 2018. [Acesso em: 13-Outubro-2018].

O histórico do desenvolvimento desse trabalho se encontra online em: https://github.com/Durfan/ufsj-aoc1-tp1.