Thales Eduardo Adair Menato – 407976
Flavia Moraes - 380048

Aula 7 – Procedures e Vetores

1) Este primeiro programa inicializa uma string com o nome “Abraham Lincoln”, carrega no registrador ecx o tamanho do nome (quantidade de caracteres). Em um primeiro loop L1, cada caractere é removido do nome e colocado no registrador eax, depois ele é colocado na pilha. Assim que todos os caracteres são colocados na pilha o loop termina e um novo loop L2 começa, onde os caracteres são removidos da pilha e adicionados novamente na memória. Como a estrutura pilha é *FILO* a string é armazenada invertida: “nlocniL maharbA”. O registrador ESP vai alterando e definindo onde está o topo da pilha, decrementando.

2) Neste código temos o main, onde está o código principal e um procedimento abaixo chamado ArraySum. O procedimento calcula a soma de todos os elementos de um array de inteiros de 32 bits e possui como entrada ESI com o endereço do array, ECX com o numero de elementos do array e gera como saída EAX com o valor calculado para a soma.
Quando ele começa o procedimento a primeira coisa a ser feita é colocar os valores na pilha do ESI e ECX, realiza a soma dentro do loop e ESI e ECX vão sendo alterados, ao terminar ele retira da pilha os valores armazenados e recupera os valores iniciais de ESI e ECX de modo que os parâmetros são preservados após passar pelo procedimento.

3) O código possui três procedimentos que são utilizados dentro do *main*. Na inicialização das variáveis o tamanho do array é definido de acordo com valor definido pra *INTEGER_COUNT* no inicio. O primeiro procedimento *PromptForIntegers* recebe como parâmetro *ECX, EDX, ESI*, ele joga para edx o endereço da string contendo “Enter a signed integer” que é utilizada assim que inicializa o loop pelo procedimento *WriteString*, abaixo temos o procedimento *ReadInt* para leitura do inteiro digitado pelo usuário no console, o *Crlf* serve para pular de linha e voltar o cursor ao início do console, o mov armazena o valor que foi armazenado em *EAX* pelo *ReadInt* para a memória no endereço de *ESI* que é o array declarado.
Após a leitura dos inteiros, temos o procedimento *ArraySum* que funciona da mesma maneira que o do exercício anteriores tirando que ele não preserva os parâmetros passados no início.
No ultimo procedimento *DisplaySum* o valor calculado é exibido na tela.

4)

```
TITLE Integer Summation Program              (Sum2.asm)

INCLUDE Irvine32.inc

TAMANHO_ARRAY = 10

.data
array DWORD  10d, 20d, 30d, 50d, 40d, 60d, 15d, 5d, 99d, 1d
somaArray DWORD ?
```

```
.code
main PROC
	call	Clrscr
	mov	esi,OFFSET array
	mov	ecx,TAMANHO_ARRAY
	call	SomaGeral
	call	WaitMsg
	exit
main ENDP

;-----------------------------------------------------
SomaGeral PROC USES esi ecx
;
; Recebe: ESI aponta para o array, ECX = tamanho do array
; Retorna:  valor da soma
;-----------------------------------------------------
	mov eax, 0
L1:	add	eax, [esi]		; store in array
	add	esi,TYPE DWORD		; next integer
	loop	L1

	mov somaArray, eax

	ret
SomaGeral ENDP

END main
```

5)