

Nº: a93280

Aluno: Miguel Ângelo Machado Martins

PL: 05

(1)(a) Depois de passarmos o arquivo "m-contaN" para a nossa diretoria, fazemos: O teste do programa. Seguem-se 2 testes, um com a cadeia de caracteres do enunciado, outro com uma cadeia de caracteres inventada por mim:

[004a93280@sc tpe7] \$./m-contaN

Introduza a cadeia de caracteres ->

1239aaswe6789g

Qual a posição inicial na cadeia de caracteres ->

3

O somatório dos dígitos na cadeia é -> 48

[004a93280@sc tpe7] \$./m-eontaN

Introduza a cadeia de caracteres ->

4326ntfs5542q

Qual a posição inicial na cadeia de caracteres ->

7

O somatório dos dígitos na cadeia é -> 16

①

(b) Abrimos o gdb:

[004a93280@sc tpe7]\$ gdb

:

(gdb) disassemble contaN

Dump Of assembler code for function contaN:

- ① 0x080483c0 <contaN+0>: pushl %ebp
- ② 0x080483e1 <contaN+1>: mov %esp, %ebp
- ③ 0x080483e3 <contaN+3>: pushl %esi
- ④ 0x080483e4 <contaN+4>: pushl %ebx
- ⑤ 0x080483e5 <contaN+5>: mov 0x8(%ebp), %esi
- ⑥ 0x080483e8 <contaN+8>: mov 0x4(%ebp), %eax
- ⑦ 0x080483eb <contaN+11>: mov (%eax,%esi,1), %dl
- ⑧ 0x080483ee <contaN+14>: xor %ebx, %ebx
- ⑨ 0x080483d0 <contaN+16>: test %dl, %dl
- ⑩ 0x080483d2 <contaN+18>: je 0x80483ee <contaN+44>
- ⑪ 0x080483d4 <contaN+20>: lea 0xffffffff(%edx), %eax
- ⑫ 0x080483d7 <contaN+23>: cmp \$0x9, %al
- ⑬ 0x080483dg <contaN+25>: ja 0x80483e2 <contaN+31>
- ⑭ 0x080483db <contaN+27>: movsb \$1, %al
- ⑮ 0x080483de <contaN+30>: lea 0xffffffff(%eax,%ebx,1), %ebx
- ⑯ 0x080483e2 <contaN+34>: inl %eax
- ⑰ 0x080483e3 <contaN+35>: mov (%eax,%esi,1), %al
- ⑱ 0x080483e6 <contaN+38>: test %al, %al
- ⑲ 0x080483e8 <contaN+40>: mov %al, %dl
- ⑳ 0x080483ea <contaN+42>: jne 0x80483d4 <contaN+20>
- ㉑ 0x080483ee <contaN+44>: mov %ebx, %eax
- ㉒ 0x080483el <contaN+46>: pop %ebx
- ㉓ 0x080483ef <contaN+47>: pop %esi
- ㉔ 0x080483f0 <contaN+48>: leave
- ㉕ 0x080483f1 <contaN+49>: ret
- ㉖ 0x080483f2 <contaN+50>: nop
- ㉗ 0x080483f3 <contaN+51>: nop

End of assembler dump

②

(e) Código assembly anotado

As linhas ① - ⑯ estão demarcadas na página 2 destas resoluções:

- ① // Inicializa a função : frame pointer antigo
- ② // Criação de novo frame pointer
- ③ // Salva guarda registo %esi
- ④ // Salva guarda registo %ebx
- ⑤ // alocação apontador para o inicio do array 5
- ⑥ // inicializa com 8 a variável de controlo do ciclo
- ⑦ // coloca o char na posição 8 de s (em %de)
- ⑧ // inicializa %ebx a 0 (que foi o registo alocado à result)
- ⑨ // teste se o char é 0 (que em ASCII é o "null")
- ⑩ // salta para o fim de s se 0
- ⑪ // subtrai h8 (ASCII: 0) a %df e põe em %al
- ⑫ // comparação de %al com 9
- ⑬ // Se o valor for > 9 (não ser dígito em ASCII) salta para 0x80h8416 <(cont. N.134> que avança em s)
- ⑭ // O char lido em %eax é estendido para 32 bits (em sinal 0 tal compl. p/2, alterado em ⑮) ⑮

- (15) // Isto é a soma de -0x3D (representação de comp. P/2 para hexa). No fundo faz a soma do dígito e acumula o resultado
- (16) // atualização do valor da posição em S
- (17) // colocação do char na nova posição S, em %eax
- (18) // verifica se char é O caractere "null" em ASCII
testa
- (19) // o char passa para %df
(%eax)
- (20) // salto para o inicio do círculo se não for
O fim de S
- (21) // devolução em %eax do result
- (22) // recuperação do registo %ebx
- (23) // recuperação do registo %esi
- (24) // recuperação do SP e FP antigo
- (25) // recuperação do IP e regresso

Identificação no código

- Os registos são atribuídos às variáveis locais result (%eax) e i (%eax)

- Os registos que são usados com os argumentos da função %esi; %eax

(4)

- a condição de teste do ciclo for
 - test $\% .\text{d}$, $\% \text{d}$
- O modo como a variável i é atualizada é $\% \text{d}$
- O código decimal correspondentes ao dígitos representados em ASCII ~~Oxfffffff~~ 0xfffffff do está em complemento para 2 e em decimal é 48
- a expressão em C que atualiza o valor de result no ciclo $\text{result} += s[i] - '0'$ $'0' (- '0')$ serve para fazer a subtração do valor ASCII

(d)

```
int contaN (char *s, int e){
    int i;
    int result=0;
    for (i=e; s[i]!='\0'; i++){
        if (s[i]>='0' && s[i]<='9')
            result += s[i]-'0';
    }
    return result;
}
```