Explicativo sobre o programa praticas_10d.s

O presente programa resolve o problema de encontrar as raízes de uma equação do segundo grau, a qual e
é definida pela seguinte equação:

ax2 + bx + c = 0

Para tanto, calcula-se o delta = b2 - 4ac e depois as raizes:

x1 = (-b + raizquad(delta))/2a e x2 = (-b - raizquad(delta))/2a

Os coeficientes a, b e c, delta e as raizes são todos números reais.

Observa-se que para raizes negativas não existe solução. Então, torna-se necessário verificar se delta < 0.

Usa-se a instrução fcom %st(1) para comparar %st91) com %st(0). Tal instrução altera o registrador de status da FPU.

No registrador de status da FPU existem 3 bits que são alterados após a execução da instrução fcom %st(1), que são: C0, C2 e C3, os quais são, da direita para esquerda, os bits 8, 10 e 14. O registrador de status pode ser visto da seguinte forma, onde B significa bit:

B15 C3 B13 B12 B11 C2 B9 C0 B7 B6 B5 B4 B3 B2 B1 B0

Tais bits podem ficar da seguinte forma:

Condição	C3	C2	C0
ST0 > %st(1)	0	0	0
ST0 < %st(1)	0	0	1
ST0 = %st(1)	1	0	0

Com isso, para verificar se delta < 0 basta colocá-lo em %st(1) e 0 em %st(0) e checar o registrador de status. Entretanto, o registrador de status deve ser antes copiado em %ax, usando a instrução fstsw %ax. Com isso, os bits que devem ser analisados ficam todos em %ah. O programa então zera os bits que não interessam usando uma instrução "and" sobre o %ah com o número 01000101 = 69. A partir disso, basta comparar %ah com 0 e usar as instruções normais de saltos (jumps) condicionais.

Para gerar o executavel, gere primeiro o objeto executando o seguinte comando:

as praticas_10d.s -o praticas_10d.o

e depois link dinamicamente com o seguinte comando:

ld praticas_10d.o -l c -dynamic-linker /lib/ld-linux.so.2 -o praticas_10d

O executavel se chamara praticas_10c, sem extensão, e para executá-lo digite:

./praticas_10d

.section .data

```
titulo:  .asciz  "\nPROGRAMA PARA ENCONTRAR RAIZES REAIS\n\n"

pedeA:              .asciz  "\nEntre com o coeficiente a => "
pedeB:              .asciz  "\nEntre com o coeficiente b => "
pedeC:              .asciz  "\nEntre com o coeficiente c => "

mostraX:      .asciz  "\nRaizes Encontradas: X1 = %.2f e X2 = %.2f \n"

msgErro:      .asciz  "\nNao Existem Raizes Reais!\n"

formato:      .asciz  "%lf"

A:            .double       0.0
B:            .double       0.0
C:            .double       0.0
DELTA:        .double       0.0
X1:           .double       0.0
X2:           .double       0.0

quatro:       .double       4.0

lixo:         .double       0.0

dois:         .double 2.0

menosum:      .double -1.0


.section .text

.globl  _start
_start:

pedeCoeficientes:

        pushl   $titulo
        call    printf

        pushl   $pedeA
        call    printf
        pushl   $A
        pushl   $formato
        call    scanf

        pushl   $pedeB
        call    printf
        pushl   $B
        pushl   $formato
        call    scanf

        pushl   $pedeC
        call    printf
        pushl   $C
        pushl   $formato
        call    scanf

        finit
```

```
calculaDelta:

        fldl    B
        fmul    %st(0), %st(0)
        fldl    quatro
        fmull   A
        fmull   C
        fsubr   %st(1), %st(0)      # st0 <= st1 - st0
        fstpl   DELTA
        fstpl   lixo

verificaDelta:

        fldz
        fldl    DELTA
        fcom                # compara st0 com st1 e altera o registrador de status
                            # nos bits C0(8), C2(10) e C3(14)

        fstsw   %ax         # copia o reg status no registrador %ax. Ambos tem
                            # 16 bits.
                            # os bits 8, 10 e 14 residem apenas no %ah
        andb    $69, %ah    # 69 = 01000101 em binario. Serve de máscara para
                            # deixar em
                            # %ah apenas os bits 8, 10 e 14

        cmpb    $1, %ah
        jz      deltaNegativo

calculaRaizes:

        fldl    DELTA
        fsqrt
        fldl    A
        fmull   dois

        fldl    B
        fsubr   %st(2), %st(0)      # st0 <= st2 - st0
        fdiv    %st(1), %st(0)      # st0 <= st0 div st1
        fstpl   X1

        fldl    menosum
        fmul    %st(2), %st(0)      # st0 <= st0 * st2
        fsubl   B
        fdiv    %st(1), %st(0)      # st0 <= st0 div st1
        fstpl   X2

        fstpl   lixo

mostraRaizes:

        fldl    X2
        subl    $8, %esp
        fstpl   (%esp)

        fldl    X1
        subl    $8, %esp
```

```
        fstpl   (%esp)

        pushl   $mostraX
        call    printf
        jmp     fim

deltaNegativo:

        pushl   $msgErro
        call    printf
fim:

        pushl   $0
        call    exit
```

Desafio 1: Declare um vetor de números em ponto flutuante. Leia o tamanho do vetor e os respectivos elementos reais. Calcule a média dos elementos e quantifique quantos estão acima e quantos estão abaixo da média.

Desafio 2: tente usar a instrução FCHS no lugar no lugar da variável "menosum"; tente alterar o trecho verificaDelta para usar a instrução sahf. Consulte o livro indicado.