CDA 3102
Fall 2024
HW #2
Gabriel Nogueras
6352093
11/3/2024

Homework #2

```
.386
.model flat, stdcall
.stack 4096
ExitProcess PROTO, dwExitCode: DWORD

.data
                ; define your variables here
                coeff_a dword 1
                coeff_b dword -25
                coeff_c dword 156
                root1 dword 0
                root2 dword 0


                ;define variables as you please
                temp_b_square dword 0           ; Stores b^2
                temp_4ac dword 0                ; stores 4ac
                discriminant dword 0            ; stores b^2 - 4ac
                sqrt_discriminant dword 0       ; stores sqrt(b^2 - 4ac)
                two_a dword 0                   ; stores 2a

.code

main PROC

        ; calculating b^2
        mov eax, coeff_b        ; eax = b
        imul eax, eax           ; int multiplication eax = b*b
        mov temp_b_square, eax   ; sets temp_b_square to eax => temp_b_square = b^2

        ; calculating 4*a*c
        mov eax, coeff_a        ; eax = a
        imul eax, 4             ; mult 4*a => eax = 4*a
        imul eax, coeff_c       ; mult new eax by c => 4*a*c
        mov temp_4ac, eax       ; set temp_4ac = eax

        ; calculating discriminant
        mov eax, temp_b_square   ; eax = b^2
        sub eax, temp_4ac       ; subtracts eax(b^2) by temp_4ac
        mov discriminant, eax    ; sets discriminant to eax

        ;check if disc is neg(no real roots)
        cmp discriminant, 0      ; compares the discriminant to 0
        jl no_real_roots         ; jumps if disc < 0, skips to the exit
```

```
        ;now we calculate using all the pre calculated parts
        mov eax, 19             ;just a placeholder for the sqrt(disc)
        mov sqrt_discriminant, eax; stores it in sqrt(disc)

        ;calculating 2*a
        mov eax, coeff_a        ; eax = a
        imul eax, 2             ; eax = 2*a
        mov two_a, eax          ; set two_a = eax

        ;calculating -b
        mov eax, coeff_b        ; eax = b
        neg eax                 ; eax = -b


        ;calculating root1 = (-b+sqrt(disc))/(2*a)
        add eax, sqrt_discriminant   ; eax = -b + sqrt(disc)
        cdq                     ; if eax is (+), edx becomes 0 and if its (-) then edx is set to -1
        idiv two_a              ; divides eax by 2a
        mov root1, eax          ; root1 = total

        ;calculate subration portion
        mov eax, coeff_b        ; eax = b
        neg eax                 ; eax = -b
        sub eax, sqrt_discriminant   ; eax = -b - sqrt(discriminant)
        cdq                     ; same as above
        idiv two_a              ; divides eax by 2a
        mov root2, eax          ; root2 = total2



        jmp end_program         ; skip

no_real_roots:
        ;case if no real roots, called above
        mov root1, 0            ; sets root1 to 0
        mov root2, 0            ; sets root2 to 0


end_program:
        INVOKE ExitProcess, 0


main ENDP
```

END main