

# Relatório trabalho “Calculadora” em Assembly

**Discentes:** Nathan Luiz Silva Oliboni e Gabriel Yudi Leite Higuchi

**Docente:** Gabriela Stein

Quanto a elaboração do trabalho, assim como descrito nas especificações, usamos funções externas da linguagem C, exatamente as especificadas:

```
extern printf
extern scanf
extern fopen
extern fprintf
extern fclose
```

Quanto às variáveis inicializadas, utilizamos:

```
section .data
    eq : db "Equação : ", 0
    ctrl : db "%f %c %f", 0
    modoAberturaArquivo : db "a", 0
    nomeArquivo : db "saida.txt", 0
    ctrlArquivo : db "%lf %c %lf = %lf", 10, 0
    ctrlArquivoNaoDisponivel : db "%lf %c %lf = funcionalidade não disponível", 10, 0
    elevadoAZero : dd 1.0, 0
    flag : db 0, 0
```

Sendo:

**eq** : O que vai ser escrito na tela na hora de pegar a equação.

**ctrl**: String de controle para pegar os dados da equação do usuário.

**ctrlArquivo**: String de controle para escrever no arquivo, caso seja um resultado aceito.

**ctrlArquivoNaoDisponivel**: String de controle para escrever no arquivo, caso seja um resultado não aceito.

**elevadoAZero**: Variável definida para caso a equação seja elevada a zero.

**flag**: Variável de controle adaptada para caso seja algum dos casos de equação não aceita.

Variáveis não inicializadas:

```
section .bss
    op1 : resd 1
    operacao: resd 1
    op2 : resd 1
    resultado : resd 1
    arq : resd 1
```

**op1**: Operando 1

**operacao**: Operação realizada (será usada na comparação e escrita no arquivo)

**op2**: Operando 2

**resultado**: Variável para armazenar o resultado da operação.

**arq**: Variável que conterá o endereço do arquivo para manipulação.

Dentro da **main** do código, fizemos o stackframe, usamos **printf** e **scanf** como ensinado em sala, decidimos utilizar o **r9b** para fazer a comparação e qual seria a operação realizada, fizemos a passagem de parâmetros utilizando os registradores de ponto flutuante **xmm0**, **xmm1** e **xmm3**(**esse exclusivamente para exponenciação**), entretanto, optei por não fazer a chamada diretamente naquela parte do código, pois por algum motivo de só Deus sabe o que rola ali, ao fazer isso estava acontecendo um erro na hora da escrita do arquivo, onde estava escrevendo duplicado (O problema estava entre o PC e a cadeira? Não sei dizer), aí fizemos **labels** somente para o **CALL** de cada função.

Dentro do label para **CALL**, é realizada para cada uma das operações a chamada da sua resolução, não utilizamos os nomes dados no arquivo de explicação e requisitos do trabalho, pois acredito que isso não infira e por motivos de estar sendo realizado a operação, não seja algo que interfira (licença poética).

Dentro de cada função, é realizado a mudança do carácter inserido pelo usuário, para o que vai ser escrito dentro do arquivo, é realizado a operação, utilizamos em sua maioria as que são realizadas diretamente (**VADDSS**, **VSUBSS**, **VMULSS**, **VDIVSS**), e armazenadas sempre em **xmm2**.

Para as funções de escrita dentro do arquivo, elas são chamadas de acordo com a solução da equação que o usuário inseriu, caso seja uma função que não tenha a solução, como especificado no pdf do trabalho, será chamada a função “**naoDisponivel**” através do call da mesma, escrevendo no arquivo que aquela função não tem uma resolução com: “funcionalidade não disponível”, caso tenha uma solução, será chamada a função “**disponivel**”, que que escreverá dentro do arquivo a equação e sua resposta.

```
disponivel:
    push rbp
    mov rbp, rsp

    movss [resultado], xmm2
    mov rax, 2
    mov rdi, qword[arq]
    mov rsi, ctrlArquivo
    cvtss2sd xmm0, [op1]
    mov rdx, [operacao]
    cvtss2sd xmm1, [op2]
    cvtss2sd xmm2, [resultado]
    call fprintf

    mov rsp, rbp
    pop rbp
    ret

naoDisponivel:
    push rbp
    mov rbp, rsp

    movss [resultado], xmm2
    mov rax, 2
    mov rdi, qword[arq]
    mov rsi, ctrlArquivoNaoDisponivel
    cvtss2sd xmm0, [op1]
    mov rdx, [operacao]
    cvtss2sd xmm1, [op2]
    call fprintf

    mov rsp, rbp
    pop rbp
```

Quanto a resolução de caso o usuário insira um operando de função errado, optamos por colocar um string de aviso no arquivo dizendo: **“Operando de entrada inválido = [operando inserido]”**

```
entradaInvalida:
    mov rdi, qword[arq]
    mov rax, 2
    mov rsi, ctrlEntradaInvalida
    mov rdx, [operacao]
    call fprintf
    jmp _fim
```

A string “ctrlEntradaInvalida” é uma variável inicializada com o que será escrito no arquivo caso o usuário entre com algum operando inválido.