Nome: Lucas Demarco Cambraia Lemos
Matrícula: 2110013

## 1ª questão

Para a tradução de foo para assembly, precisei guardar o valor de %xmm1 no registrador auxiliar %xmm2. Tentei rodar tirando o protótipo da função do arquivo .c e retornou 0. Acredito que seja um retorno padrão para quando isso aconteça.

```
1   .text
2   .globl  foo1
3   foo1:
4     cvtsd2ss %xmm0, %xmm0
5     movss  %xmm1, %xmm2
6     addss  %xmm0, %xmm2
7     subss  %xmm1, %xmm0
8     mulss  %xmm2, %xmm0
9     ret
10
```

Saída do código da questão 1 para **a** = 1 e **b** = 2:

```
[brava.grad.inf.puc-rio.br:~/inf1018/2023/LAB15] ./lab15-ex1
3.000000
[brava.grad.inf.puc-rio.br:~/inf1018/2023/LAB15]
```

## 2ª questão

Para a tradução da função foo1 para assembly, foi necessário guardar o segundo parâmetro (%xmm1) na pilha, pois este valor poderia ser perdido na chamada da função sin. Segue abaixo o código assembly para a tradução da função.

```
1   .text
2   .globl  foo2
3   foo2:
4     pushq  %rbp
5     movq  %rsp, %rbp
6     subq  $16, %rsp
7     cvtss2sd  %xmm0, %xmm0
8     cvtss2sd  %xmm1, %xmm1
9     movsd  %xmm1, -8(%rbp)
10    call  sin
11    movsd  -8(%rbp), %xmm1
12    addsd  %xmm1, %xmm0
13    cvtsd2ss %xmm0, %xmm0
14    leave
15    ret
16
```

Saída do código da questão 2, sendo o primeiro parâmetro pi/2 e o segundo parâmetro 1:

```
[geriba.grad.inf.puc-rio.br:~/inf1018/2023/LAB15] ./lab15-ex2
2.000000
[geriba.grad.inf.puc-rio.br:~/inf1018/2023/LAB15]
```

**3ª questão**

Para a tradução dessa função para assembly, foi necessário realizar dois salvamentos na pilha. Inicialmente, foi necessário salvar %xmm1 na pilha, pois na chamada da função sin, esse valor pode ser perdido. Assim que obtive o valor de sin(a), salvei na pilha, pois poderia perdê-lo ao chamar a função cos. Segue o código assembly abaixo:

```
1   .text
2   .globl  foo3
3   foo3:
4     pushq  %rbp
5     movq  %rsp, %rbp
6     subq  $16, %rsp
7     cvtss2sd  %xmm0, %xmm0
8     cvtss2sd  %xmm1, %xmm1
9     movsd  %xmm1, -8(%rbp)
10    call  sin
11    movsd  %xmm0, -16(%rbp)
12    movsd  -8(%rbp), %xmm0
13    call  cos
14    addsd  -16(%rbp), %xmm0
15    leave
16    ret
```

Saída do código da questão 3. (Primeiro parâmetro = pi/2; Segundo parâmetro = 0):

```
[geriba.grad.inf.puc-rio.br:~/inf1018/2023/LAB15] gcc -Wall -o lab15-ex3
 foo3.s mainfoo3.c -lm
[geriba.grad.inf.puc-rio.br:~/inf1018/2023/LAB15] ./lab15-ex3
2.000000
```

]

**4ª questão**

Para a tradução da função *foo* da 4ª questão, foi necessário guardar 3 valores na pilha: os dois parâmetros da função (%rdi e %esi), além do valor de *double r* (%xmm1). Segue o código em assembly dessa função:

```
1   .text
2   .globl  foo4
3   foo4:
4     pushq  %rbp
5     movq  %rsp, %rbp
6     subq  $32, %rsp
7     movl  $0, %r8d   /* int i = 0 */
8     movl  $0, %r9d
9     cvtsi2sd  %r9d, %xmm1     /* double r = 0.0 */
10
11  for:
12    cmpl  %esi, %r8d
13    jge  end_for
14    movsd  %xmm1, -16(%rbp)
15    movq  %rdi, -24(%rbp)
16    movl  %esi, -28(%rbp)
17    movsd  (%rdi), %xmm0
18    call  sin
19    movsd  -16(%rbp), %xmm1
20    movq  -24(%rbp), %rdi
21    movl  -28(%rbp), %esi
22    addsd  %xmm0, %xmm1
23    addq  $8, %rdi
24    incl  %r8d
25    jmp  for
26
27  end_for:
28    movsd  %xmm1, %xmm0
29    leave
30    ret
```

Saída do código da questão 4, passando o vetor *double a[ ]* = {pi/2, pi/2} e *int n* = 1:

```
(base) puc@notepuc07:~/Documentos/Pessoal/inf1018/inf1018-software-basico/LAB15$ gcc -Wall -o lab15-ex4 fo
o4.s mainfoo4.c -lm
(base) puc@notepuc07:~/Documentos/Pessoal/inf1018/inf1018-software-basico/LAB15$ ./lab15-ex4
2.000000
```