

Aluno: Lucas de Medeiros Nunes Fernandes
Matrícula: 117110210

1. Conversões:

Decimal	Binário inteiro complemento de 2	Binário Real Ponto Flutuante (IEEE 754) (polarização = 011111112 =12710).
+23	000000000000000000000000 0000010111	0x41b80000
0	000000000000000000000000 0000000000	0x00000000
NAN	111111111111111111111111 1111111000	0xFFFFFFFF8

2.

- Como o número 0.1 não tem uma representação finita em binário, então seguindo o padrão IEEE-754, sua representação tem um valor um pouco maior que 0.1, como foi somada repetidas vezes o valor 0.1, e o tipo da variável é **float**, o pequeno erro na representação vai se tornando perceptível.
- Como o tipo **double** tem tamanho maior de casas decimais que **float** para a representação de números reais, o erro se acumula de maneira menos perceptível.

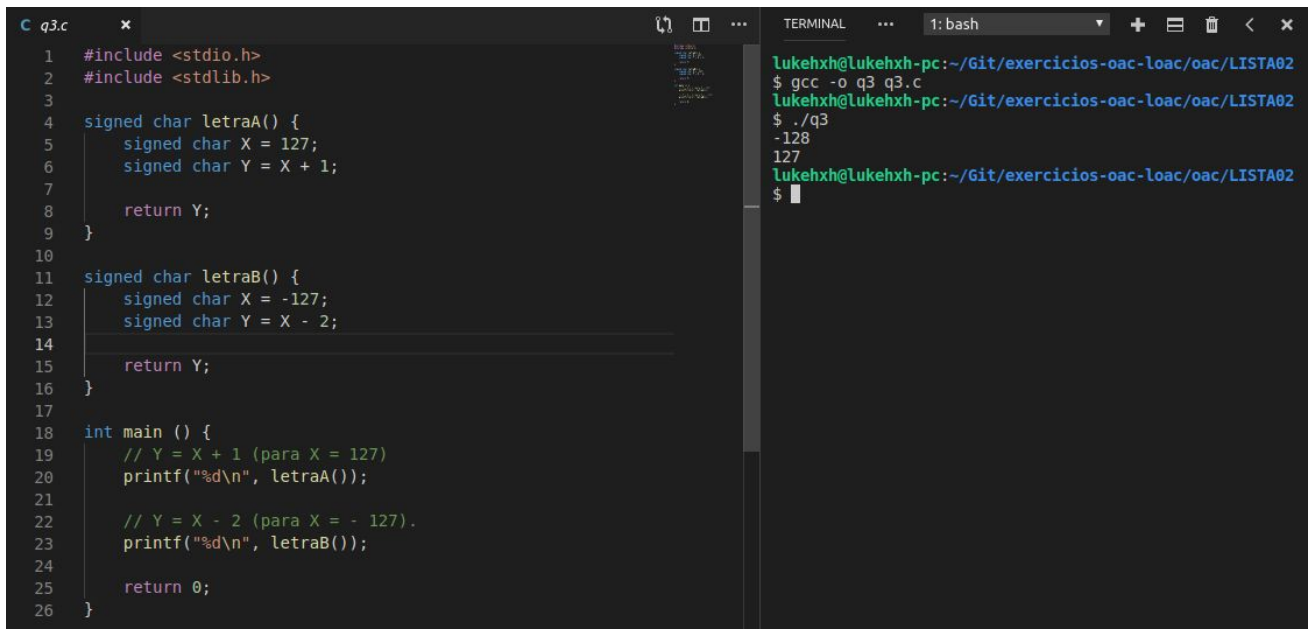
Ilustração das respostas aos itens A e B (tela de execução do código):

```
c q2.c x
1 #include <stdio.h>
2 #include <stdlib.h>
3
4 float letraA() {
5     float Y = 0;
6     float X = 0.1;
7
8     for(int i = 0; i <= 99; i++) {
9         Y += X;
10    }
11
12    return Y;
13 }
14
15 double letraB() {
16     double Y = 0;
17     double X = 0.1;
18     for(int i = 0; i <= 99; i++) {
19         Y += X;
20     }
21     return Y;
22 }
23
24 int main () {
25     // Definir X e Y como variáveis do tipo Float.
26     printf("%f\n", letraA());
27
28     // Definir X e Y como variáveis do tipo Double.
29     printf("%f\n", letraB());
30
31     return 0;
32 }
```

```
TERMINAL ... 1: bash
lukehxh@lukehxh-pc:~/Git/exercicios-oac-loac/oac/LISTA02
$ gcc -o q2 q2.c
lukehxh@lukehxh-pc:~/Git/exercicios-oac-loac/oac/LISTA02
$ ./q2
10.000002
10.000000
lukehxh@lukehxh-pc:~/Git/exercicios-oac-loac/oac/LISTA02
$
```

3.

- a. O tipo **signed char** em C tem seus valores dentro de um intervalo de -128 a 127, quando há overflow (qualquer valor acima de 127), o valor não é mostrado nem tratado na execução do programa, fazendo com que o valor “dê a volta” e vá para -128.
- b. De maneira similar, o underflow também não é mostrado nem tratado durante a execução do programa, fazendo com que o valor também “dê a volta” e vá para 127.



```
1 #include <stdio.h>
2 #include <stdlib.h>
3
4 signed char letraA() {
5     signed char X = 127;
6     signed char Y = X + 1;
7
8     return Y;
9 }
10
11 signed char letraB() {
12     signed char X = -127;
13     signed char Y = X - 2;
14
15     return Y;
16 }
17
18 int main () {
19     // Y = X + 1 (para X = 127)
20     printf("%d\n", letraA());
21
22     // Y = X - 2 (para X = -127)
23     printf("%d\n", letraB());
24
25     return 0;
26 }
```

```
lukehvh@lukehvh-pc:~/Git/exercicios-oac-loac/oac/LISTA02
$ gcc -o q3 q3.c
lukehvh@lukehvh-pc:~/Git/exercicios-oac-loac/oac/LISTA02
$ ./q3
-128
127
lukehvh@lukehvh-pc:~/Git/exercicios-oac-loac/oac/LISTA02
$
```