

Lista de exercícios 3

Biblioteca math.h: Funções pré-definidas de arredondamento

Resumo

Este documento apresenta uma de resolução de exercício em C++, envolvendo funções de arredondamento e o uso da biblioteca math.h, desenvolvidos na disciplina de Algoritmos e Técnicas de Programação. O exercício tem como objetivo fornecer a prática necessária para compreender e aplicar os conceitos de arredondamento, fundamentais para o entendimento da disciplina.

Exercício 1

Assista ao [vídeo do Youtube](#) sobre funções aritméticas pré-definidas de arredondamento e truncamento, então faça um relatório explicando cada uma destas funções. Use exemplos de códigos e imagens (print screens) das execuções dos programas para complementar suas explicações.

Resolução

A função `floor()` é usada para arredondar um valor real n para o menor inteiro próximo de n . A função receberá o parâmetro de um número real e retornará o arredondamento deste número.

Podemos dizer que $\text{floor}(n.\underbrace{c_1 c_2 \cdots c_m}_{m \text{ decimais}}) = \lfloor n \rfloor$ tal que $\lfloor n \rfloor < n$.

Exemplo:

```
1 #include <iostream>
2 #include <math.h>
3 using namespace std;
4 int main(){
5
6     float valor;
7     cout << "Digite um numero real: ";
8     cin >> valor;
9     cout << floor(valor);
10
11     return 0;
12 }
```



Digite um numero real: 5.3

5

Process exited after 2.213 seconds with return value 0

Pressione qualquer tecla para continuar. . .

Também podemos fazer o arredondamento sem usar a biblioteca `math.h`:

```
1 #include <iostream>
2 using namespace std;
3 int main(){
4     float valor;
5     int arredondamentoBaixo;
6
7     cout << "Digite um numero real: ";
8     cin >> valor;
9     arredondamentoBaixo = valor;
10
11     cout << arredondamentoBaixo;
12     return 0;
13 }
```



Digite um numero real: 4.5

4

Process exited after 4.363 seconds with return value 0

Pressione qualquer tecla para continuar. . .

A função `ceil()` é usada para arredondar um valor real n para o maior inteiro próximo de n . A função receberá o parâmetro de um número real e retornará o arredondamento deste número.

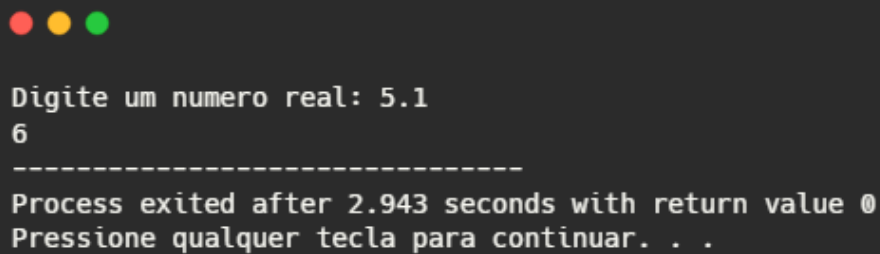
Podemos dizer que $\text{ceil}(n, \underbrace{c_1 c_2 \cdots c_m}_{m \text{ decimais}}) = \lceil n \rceil$ tal que $\lceil n \rceil > n$.

Exemplo:

```

1 #include <iostream>
2 #include <math.h>
3 using namespace std;
4 int main(){
5
6     float valor;
7     cout << "Digite um numero real: ";
8     cin >> valor;
9     cout << ceil(valor);
10
11     return 0;
12 }

```



```

Digite um numero real: 5.1
6
-----
Process exited after 2.943 seconds with return value 0
Pressione qualquer tecla para continuar. . .

```

Também é possível escrever a função `ceil()` sem o uso da biblioteca `math.h`:

```

1 #include <iostream>
2 using namespace std;
3 int main(){
4
5     float valor;
6     int arredondamentoAlto;
7
8     cout << "Digite um numero real: ";
9     cin >> valor;
10
11     arredondamentoAlto = valor + 1;
12
13     cout << arredondamentoAlto;
14     return 0;
15 }

```

```

1 2 3
Digite um numero real: 1.2
2
-----
Process exited after 3.935 seconds with return value 0
Pressione qualquer tecla para continuar. . .

```

A função `round()` é usada para arredondar um valor real n para o seu piso ou teto, a função avalia se o valor decimal é maior ou igual a 0.5, caso seja maior, a função fará o arredondamento para o teto (`ceil()`), caso seja menor, o arredondamento será para o piso (`floor()`). A função receberá o parâmetro de um número real e retornará o arredondamento adequado para o número.

Assim:

$$\text{round}(n) = \begin{cases} \text{ceil}(n), & \text{se casas decimais de } n \geq 0.5 \\ \text{floor}(n), & \text{se casas decimais de } n < 0.5 \end{cases}$$

Exemplo:

```

1 #include <iostream>
2 #include <math.h>
3 using namespace std;
4 int main(){
5
6     float valor;
7
8     cout << "Digite um numero real: ";
9     cin >> valor;
10
11     cout << round(valor);
12
13     return 0;
14 }

```

Para casas decimais de $n < 0.5$:



Digite um numero real: 1.2

1

Process exited after 23.69 seconds with return value 0

Pressione qualquer tecla para continuar. . .

Para casas decimais de $n = 0.5$:



Digite um numero real: 1.5

2

Process exited after 3.875 seconds with return value 0

Pressione qualquer tecla para continuar. . .

Para casas decimais de $n > 0.5$:



Digite um numero real: 1.8

2

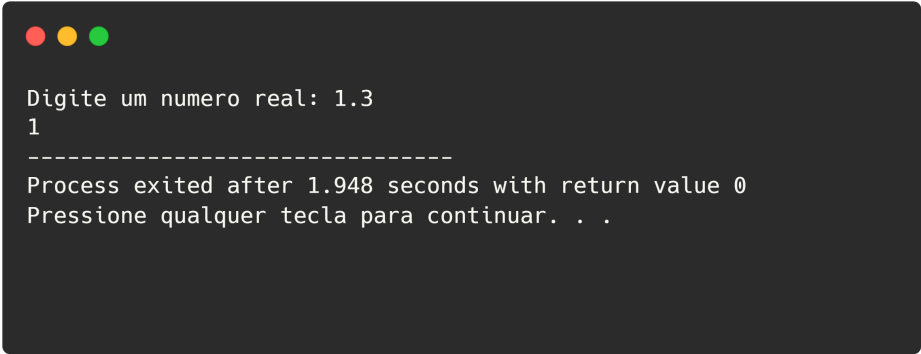
Process exited after 2.907 seconds with return value 0

Pressione qualquer tecla para continuar. . .

Também podemos escrever essa função sem usar a biblioteca `math.h`:

```
1 #include <iostream>
2 using namespace std;
3 int main(){
4
5     float valor;
6
7     cout << "Digite um numero real: ";
8     cin >> valor;
9
10    if (valor - int(valor) >= 0.5){
11        cout << int(valor) + 1;
12    }
13    else {
14        cout << int(valor);
15    }
16
17    return 0;
18 }
```

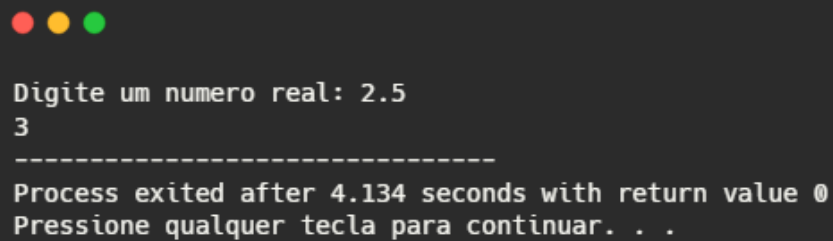
Para casas decimais de $n < 0.5$:



```

Digite um numero real: 1.3
1
-----
Process exited after 1.948 seconds with return value 0
Pressione qualquer tecla para continuar. . .
```

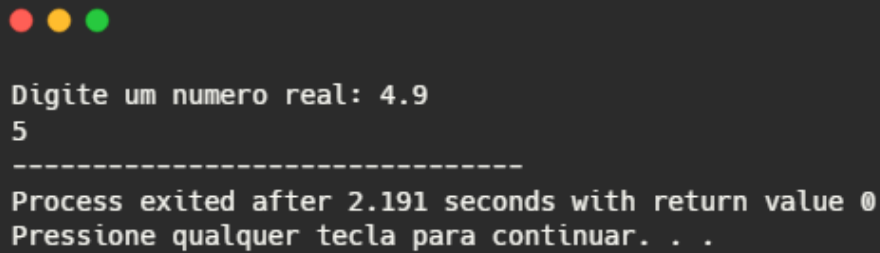
Para casas decimais de $n = 0.5$:



```

Digite um numero real: 2.5
3
-----
Process exited after 4.134 seconds with return value 0
Pressione qualquer tecla para continuar. . .
```

Para casas decimais de $n > 0.5$:



```

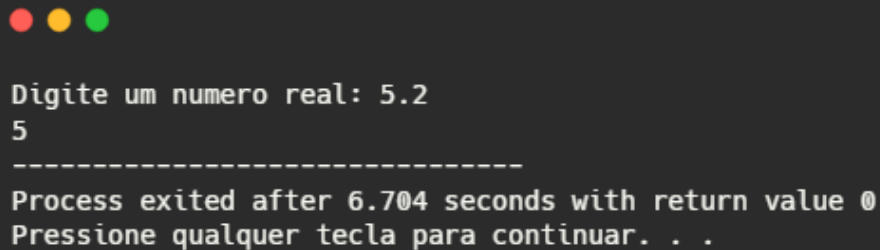
Digite um numero real: 4.9
5
-----
Process exited after 2.191 seconds with return value 0
Pressione qualquer tecla para continuar. . .

```

A função `trunc()` remove as casas decimais. Recebe com parâmetro um número real n e retorna o número após o truncamento. Exemplo:

```

1 #include <iostream>
2 #include <math.h>
3 using namespace std;
4 int main(){
5
6     float valor;
7
8     cout << "Digite um numero real: ";
9     cin >> valor;
10
11     cout << trunc(valor);
12
13     return 0;
14 }
```



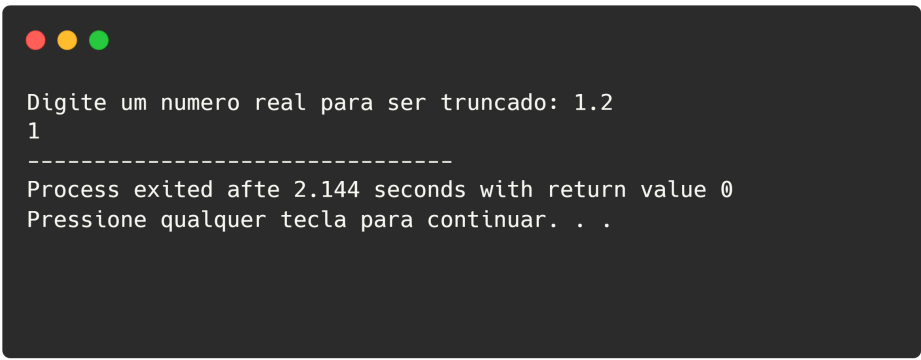
```

Digite um numero real: 5.2
5
-----
Process exited after 6.704 seconds with return value 0
Pressione qualquer tecla para continuar. . .

```

Podemos realizar o truncamento sem a biblioteca `math.h`:

```
1 #include <iostream>
2 using namespace std;
3 int main(){
4
5     float valor;
6     int truncamento;
7
8     cout << "Digite um valor real para ser truncado: ";
9     cin >> valor;
10
11     truncamento = valor;
12
13     cout << truncamento;
14     return 0;
15 }
```

A terminal window with a dark background and three colored window control buttons (red, yellow, green) in the top-left corner. The text inside the terminal shows the program's execution: a prompt to enter a real number, the input '1.2', the output '1', a separator line, and a message indicating the process exited after 2.144 seconds with a return value of 0, followed by a prompt to press any key to continue.

● ● ●

Digite um numero real para ser truncado: 1.2
1

Process exited afte 2.144 seconds with return value 0
Pressione qualquer tecla para continuar. . .