|  |
| --- |
| Project 2  Biblioteca EASYSTD e POO em C++  Feito por: João Roque e Licínio Feliciano |



Conteúdo

[Introdução 2](#__RefHeading___Toc505_2479917017)

[Implementação e Instruções de Uso 3](#__RefHeading___Toc286_93532997)

[Biblioteca EASY\_STD 4](#__RefHeading___Toc282_93532997)

[Biblioteca EASY\_RANDOM 6](#__RefHeading___Toc284_93532997)

[Gestão de Viaturas 7](#__RefHeading___Toc284_93532997_Copy_1)

[Conclusão 8](#__RefHeading___Toc511_2479917017)

[Conclusão 9](#__RefHeading___Toc294_93532997)

# **Introdução**

Este projecto tem como objectivo desenvolver uma biblioteca pessoal de utilitários em C/C++, com o foco de consolidar os conhecimentos de programação nesta linguagem. A biblioteca, denominada EasySTD, visa simplificar o desenvolvimento em C++, oferecendo mecanismos que, embora não estejam presentes na biblioteca padrão do C++, são necessários para diversas aplicações. A biblioteca abordará três temas principais: Utilitários Gerais, Texto e Aleatoriedade. Todas as definições desta biblioteca serão integradas no namespace easy\_std.

A biblioteca EasySTD é projectada para ser útil, simples, fácil de implementar e legível, priorizando a utilidade e a simplicidade sobre o desempenho. Isso é especialmente relevante em cenários de programação de sistemas, onde o bom desempenho e a gestão dos recursos do computador são requisitos importantes. No entanto, na biblioteca EasySTD, o bom desempenho é considerado um requisito secundário.

# Implementação e Instruções de Uso

**Compilacao:**

Para compilar o programa, use um compilador C++ com suporte para o padrão C++20.

No Linux/Unix: g++ -std=c++20 teste\_easy\_\*.cpp -o teste

No Windows: g++ -std=c++20 teste\_easy\_\*.cpp -o teste.exe

- *Substitua \* pela referência do teste (utils, text, random).*

**Execução:**

Execute o ficheiro compilado: ./teste (Linux/MacOS) ou teste.exe (Windows).

Siga as instruções no ecrã para testar as funções.

**Utilização:**

O programa irá correr vários *checks*, uns interactivos e outros automáticos, de modo a demonstar o funcionamento de cada função.

**Encerrar o programa:**

Para encerrar o programa, basta fazer a combinação de teclas **CTRL+D, CTRL+C**, ou correr todos os checks até ao fim.

# Biblioteca EASY\_STD

**is\_white\_space(ch) -** Devolve **true** se o caractere for um caractere de espaçamento. (ex: ' ', '\n', '\t' ou '\r').

**is\_white\_space(str) -** Devolve **true** se todos os caractere da string forem de espaçamento. Deve utilizar **is\_white\_space(ch)**.

**is\_digit(ch) -** Devolve **true** se o caractere for um dígito decimal.

**is\_digit(str) -** Devolve **true** se todos os caractere da *string* forem dígitos. Deve utilizar **is\_digit(ch)**.

**to\_lower(ch) -** Converte um caractere (letra) maiúscula para minúscula.

**to\_lower(str) -** Converte um caractere (letra) maiúscula para minúscula. Esta

função modifica a *string* e devolve-a para que possa ser utilizada em expressões

do tipo *string*.

**trim\_left(str) -** Remove os caracteres de espaçamento à esquerda da *string*. A função modifica a *string* e devolve-a para que possa ser utilizada.

**trim\_right(str) -** Remove os caracteres de espaçamento à direita da *string*. A função modifica a *string* e devolve-a para que possa ser utilizada.

**trim(str) -** Remove os caracteres de espaçamento à esquerda e direita da *string*. A função modifica a *string* e devolve-a para que possa ser utilizada.

**reverse(str)-** Inverte a *string*. A função modifica a *string* e devolve-a para que possa ser utilizada. (ex: antonio jose => esoj oinotna)

**Reversed(str) -** Devolve uma cópia invertida da *string* com os caracteres por ordem inversa. (ex: antonio jose => oinotna esoj )

**replace (STRING, CH1, CH2, [START], [END]) -** Substitui todas as ocorrências do caractere **CH1** pelo caractere **CH2**. A função modifica a *string* e devolve-a para que possa ser utilizada.

Opções:

**STRING** - A *string* que vai sofrer alterações.

**CH1 –** Caractere a ser modificado.

**CH2** – Caractere que vai substituir **CH1**.

**START (int)** – A posiçao inicial para substituições [default: 0].

**END (int) –** A posição final para substituições [default: -1, ou seja, último caractere].

**replace (STRING, SUBSTR1, SUBSTR2, [START], [END]) –** Sunstitui todas as ocorrências da **substring1** pela **substring2**. A função modifica a *string* e devolve-a para que possa ser utilizada.

Opções:

**STRING -** A *string* que vai sofrer alterações.

**SUBSTR1 -** *substring* a ser modificada.

**SUBSTR2 -** A *substring* que vai substituir **SUBSTR1.**

**START (int)-** A posiçao inicial para substituições [default: 0].

**END (int)-** A posição final para substituições [default: -1, ou seja, última substring].

**split (STRING, DELIM) -** Separa a *string* em partes, usando como delimitador a variavel DELIM.

Opções:

**STRING –** *string* a separar. (ex: "ABC.DEF.GHI")

**DELIM –** delimitador a usar para a separação. (ex: split("ABC.DEF.GHI", ".") => {"ABC", "DEF", "GHI"})

**join {PARTS, DELIM} –** Junta todas as *strings*, usando como delimitador a variavel **DELIM.**

Opções:

**PARTS -** *vector* de *strings* a juntar. (ex: {"ABC”, “DEF”, “GHI"})

**DELIM –** delimitador a usar para junção. (ex: join{"ABC, “DEF”, “GHI", "."} => "ABC.DEF.GHI")

# **Biblioteca EASY\_RANDOM**

**randint (int a, int b) -** Devolve um *int* entre *int* **a** e *int* **b** ínclusive.

**random(double a, double b) -** Devolve um *double* entre *double* **a** e *double* **b** ínclusive. Por omissão, a = 0.0 e b = 1.0.

**seed(int value) -** Fornece um novo valor semente para o gerador de número aleatório rand\_gen.

**choice(const Seq& seq) -** Escolhe aleatoriamente um elemento da sequência *seq*.

**shuffle\_(Seq& seq) -** Mistura aleatóriamente os elementos da sequência *seq*.

**sample(const Seq& seq, int n) -** Selecciona uma amostra de *n* valores da sequência *seq*. Os resultados são devolvidos num vector do tipo da variável de retorno ou do tipo de chamamento.

# **Gestão de Viaturas**

# Conclusão

Ao concluir este projecto, é possível afirmar que a programação em C/C++ é uma ferramenta poderosa e versátil. A criação da biblioteca EasySTD, focada em simplificar e facilitar o desenvolvimento em C++, demonstra a importância de ter ferramentas que integrem e simplifiquem tarefas comuns, permitindo que os desenvolvedores se concentrem na lógica do programa e na solução de problemas específicos.

O desenvolvimento de um programa para a gestão de um catálogo de viaturas de um stand automóvel, além de ser um exemplo prático da aplicação da biblioteca EasySTD, também ilustra a capacidade da linguagem C++ de ser utilizada em contextos diversos, desde aplicações simples até sistemas mais complexos. Este projecto, portanto, não apenas consolida os conhecimentos de programação em C/C++, mas também destaca a importância de criar bibliotecas personalizadas que atendam às necessidades específicas dos projectos, contribuindo para a eficiência e produtividade do desenvolvedor.