|  |
| --- |
| Project 1 Biblioteca EASYSTD e POO em C++  Feito por: João Roque e Licínio Feliciano |



Table of Contents

[Project 1 1](#__RefHeading___Toc280_93532997)

[Biblioteca EASY\_STD 2](#__RefHeading___Toc282_93532997)

[Biblioteca EASY\_RANDOM 4](#__RefHeading___Toc284_93532997)

[Gestão de Viaturas 5](#__RefHeading___Toc284_93532997_Copy_1)

[Implementação e Instruções de Uso 6](#__RefHeading___Toc286_93532997)

[Efeito em V 7](#__RefHeading___Toc288_93532997)

[Deslizante 8](#__RefHeading___Toc290_93532997)

[EXCEL 9](#__RefHeading___Toc292_93532997)

[Conclusão 10](#__RefHeading___Toc294_93532997)

Introdução

Biblioteca EASYSTD e POO em C++ é um conjunto de bibliotecas criadas para facilitar a utilização da linguagem C++.

# Biblioteca EASY\_STD

**is\_white\_space(ch) -** Devolve **true** se o caractere for um caractere de espaçamento, ou seja, se for um dos seguintes:

' ', '\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**

# 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 efeito.cpp -o efeito

No Windows: g++ -std=c++20 efeito.cpp -o efeito.exe

**Execução:**

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

Siga as instruções no ecra para escolher diferentes efeitos de exibição.

**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 checks até ao fim.

## Conclusão

O Programa “Efeitos de texto” apresenta efeitos de exibição de texto por meio de código C++. Este programa serve para experimentação visual e fornece uma compreensao sobre programação em C++ e manipulação do terminal.