6/09/2025

METODI

1. empty()

Definizione: Metodo della classe std::string che controlla se una stringa è vuota.

Sintassi: bool str.empty() const;

Descrizione: Restituisce true se la stringa non contiene caratteri. Restituisce false se la stringa ha almeno un carattere. È equivalente a verificare se str.size() == 0. Non modifica la stringa.

2. open()

Definizione: Metodo della classe std::ifstream o std::ofstream per aprire un file.

Sintassi: void open(const char* filename, ios_base::openmode mode = ios_base::in);

Descrizione: Apre un file con il nome specificato. Il secondo parametro (mode) indica il modo di apertura (in, out, app, ecc.). Se il file non può essere aperto (inesistente, mancanza permessi), lo stream entra in stato di errore. Non restituisce un valore: per controllare il successo si usa is_open().

3. close()

Definizione: Metodo di std::ifstream o std::ofstream che chiude un file aperto.

Sintassi: void close();

Descrizione: Interrompe l'associazione tra lo stream e il file sul disco. È buona pratica chiamarlo sempre dopo aver finito di leggere/scrivere. Se lo stream viene distrutto senza close(), il file viene chiuso automaticamente, ma chiamarlo esplicitamente è più chiaro.

4. is_open()

Definizione: Metodo di std::ifstream e std::ofstream che verifica se un file è aperto correttamente.

Sintassi: bool is_open() const;

Descrizione: Restituisce true se il file è stato aperto con successo. Restituisce false se non è stato possibile aprirlo. Serve per controllare subito dopo open() se lo stream è valido.

5. getline()

Definizione: Funzione di libreria che legge una riga da uno stream di input (come std::ifstream o std::cin) e la memorizza in una stringa.

Sintassi: std::getline(std::istream& is, std::string& str);

Descrizione: Legge caratteri da uno stream fino al primo carattere di nuova riga ('\n') o fino alla fine del file. Memorizza i caratteri letti in str. Non include il carattere di nuova riga nella stringa. Se il file è finito (EOF), lo stream entra in stato di errore.

6. erase()

Definizione: Metodo di std::string che rimuove una parte della stringa.

Sintassi: string& str.erase(size_t pos = 0, size_t len = npos);

Descrizione: Cancella len caratteri a partire dalla posizione pos. Se len non è specificato, cancella fino alla fine della stringa. Modifica la stringa originale.

Esempio:

```
std::string s = "Hello World";
s.erase(5); // s diventa "Hello"
```

7. insert()

Definizione: Metodo di std::string che inserisce una sottostringa o un carattere in una posizione specificata.

Sintassi:

string& str.insert(size_t pos, const string& str2);

Descrizione: Inserisce str2 nella stringa originale a partire dalla posizione pos. Se pos è maggiore della lunghezza della stringa, viene generata un'eccezione (out_of_range). Modifica la stringa originale.

Esempio:

```
std::string s = "Hello World";
s.insert(5, ",");
// Risultato: "Hello, World"
```

8. length()

Definizione: Metodo di std::string che restituisce il numero di caratteri contenuti nella stringa.

Sintassi: size_t str.length() const;

Descrizione: Ritorna la lunghezza attuale della stringa. È equivalente a str.size(). Non conta il terminatore '\0', ma solo i caratteri reali. Non modifica la stringa.

9. find()

Definizione: Metodo di std::string che cerca una sottostringa all'interno della stringa.

Sintassi: size_t str.find(const string& sub, size_t pos = 0) const;

Descrizione: Cerca la sottostringa sub a partire dalla posizione pos. Restituisce la posizione della **prima occorrenza** trovata. Se non trova nulla, restituisce std::string::npos (che corrisponde a -1 non segnato). Non modifica la stringa.

Esempio:

```
std::string s = "Hello World";
size_t pos = s.find("World"); // restituisce 6
```