Strings C++

Prof. Alexandro M. S. Adário

Strings C-style: O Básico (Legado)

- Strings = char[] + '\0'
- Herdadas da linguagem C

```
char s[] = "Cplus"; // 6 bytes (inclui '\0')
```

Operações e Riscos do C-style

- strcpy(dest, src) \rightarrow 1 buffer overflow
- strcat(dest, src) \rightarrow 1 cópia ilimitada
- $strlen(s) \rightarrow lenta$, tempo linear



Sempre verificar tamanho de buffers!

Segurança: Problemas da Não Delimitação

- Cópias não delimitadas
- Falha na terminação nula

```
char buf[10];
cin >> buf; //Se entrada > 9 caracteres →
out-of-bounds
```

std::string: A Solução Segura

- Classe padrão <string> → RAII
- Tamanho dinâmico e seguro

```
std::string s1 = "Ola";
s1 += " Mundo";
```

Comprimento: s1.size() ou s1.length()

Interoperabilidade com C: c_str()

- Passa std::string para APIs C legadas (const char*)
- Retorna ponteiro constante terminado em '\0'

```
minha_string.c_str();
```

```
↑ data() ≈ c_str() (C++11+)
```

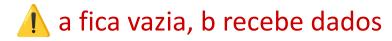
Otimização SSO (Small String Optimization)

- Strings curtas (~15 chars) armazenadas no objeto
- Evita alocação na heap
- Benefício: rápido, melhor localidade de cache

Semântica de Movimentação

- Transfere recursos sem cópia profunda
- std::move() habilita a movimentação

```
std::string a = "texto grande";
std::string b = std::move(a);
```



std::string_view (C++17)

- Visualização de leitura, leve, sem cópia
- Estrutura: ponteiro + tamanho

```
void imprime(std::string_view texto) {
    std::cout << texto;
}</pre>
```



Não deve superar a vida da string original (dangling view)

Buscas e Testes Simplificados (C++20/23)

- starts_with("abc") / ends_with("xyz") → C++20
- contains("palavra") → C++23

```
std::string url = "https://site.com";
if (url.contains("https")) { /* ... */ }
```

Formatação Segura: std::format (C++20)

- Substitui placeholders { }
- Mais seguro e eficiente que printf/sprintf

```
std::string s = std::format("Nome: {}, ID: {}",
name, id);
```

Melhores Práticas (Resumo)

- Prefira std::string a C-style strings inseguras
- Use std::string_view para funções read-only
- Gerencie recursos automaticamente (std::string, std::unique_ptr)