一、字符串的处理（String Processing）

什么是字符串（String）？

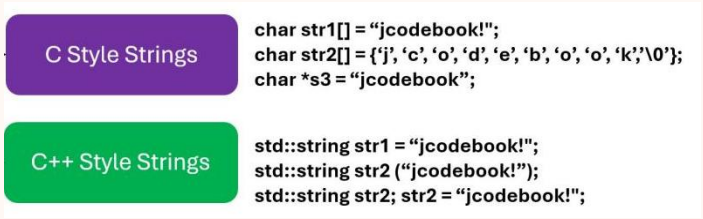
字符串（String）是由**一串字符（Characters）**组成的数据类型，常用来存储名字、句子、文字等。

字符串就像你写的一句话，比如：

"Hello!"

"我爱编程"

"小明 123"



字符串是用**英文双引号 ""** 括起来的。

（一）字符数组与相关函数（Char Array & Functions）

☑什么是字符数组?

字符数组是最早的字符串表达方式，用**数组来存储多个字符（char）**。

以 char 类型表示，每个字符单独存储，最后**必须有一个 '\0'（空字符）** 表示字符串的结束。

示例代码：

```
char name[6] = {'A', 'l', 'i', 'c', 'e', '\0'};
cout << name; // 输出 Alice
```

也可以直接这样写：

```
char name[] = "Alice";
```

就像你拼一个名字，每个字母占一个格子，最后加个“空白块”表示名字拼完了。

i	0	1	2	3	4	5
name[]	‘A’	‘l’	‘i’	‘c’	‘e’	‘\0’

☑ 常用字符数组函数（需要 #include <cstring>）

1. strlen()

作用：像量身高一样量字符串有多长（不算结尾的\0）

```
#include <cstring>
char fruit[] = "apple";
int len = strlen(fruit);   // len=5
cout << "苹果有" << len << "个字母";
```

2. strcpy()

作用：复制字符串（像复印机）。注意：要确保目标数组足够大，否则会出错。

```
char source[] = "Hello";
char target[10];
strcpy(target, source);   // 把 source 的内容复制到 target
cout << target;   // 输出"Hello"
```

3. strcat()

作用：连接两个字符串（像拼积木）。注意：目标数组必须足够大，能容纳连接后的字符串。

```
char s1[20] = "I love ";
char s2[] = "C++!";
strcat(s1, s2);   // 把 s2 拼接到 s1 后面
cout << s1;   // 输出"I love C++!"
```

4. strcmp()

作用：比较两个字符串（像比赛排名）。返回 0 表示相等，负数表示第一个小于第二个，正数表示第一个大于第二个。

```
char word1[] = "apple";
```

```
char word2[] = "banana";
if(strcmp(word1, word2) < 0) {
    cout << "apple 在 banana 前面";  // 因为 a 在 b 前面
}
```

5. strstr()

作用：在大字符串里找小字符串（像玩寻宝游戏）。如果找到，返回第一次出现的位置（指针）；没找到返回空指针（NULL）。

```
char text[] = "CSP-J 竞赛很有趣";
char *pos = strstr(text, "竞赛");  // 找"竞赛"的位置
if(pos != nullptr) {
    cout << "在位置" << (pos-text) << "找到了！";
}
```

6. strchr()

作用：在字符串中查找一个字符（像在单词里找字母）。返回字符第一次出现的位置（指针），没找到返回 NULL。

```
char str[] = "programming";
char *p = strchr(str, 'm');  // 找字母 m
if(p != nullptr) {
    cout << "m 在位置" << (p-str);  // 输出位置 6
}
```

7. memset()

作用：把一段内存块设成同一个值（像用颜料刷墙）。

```
char line[10];
memset(line, '-', 5);  // 前 5 个字符变成'-'
```

```
line[5] = '\0';            // 记得加结束符
cout << line;              // 输出"-----"
```

8. cin.getline()

作用：读取整行输入（包含空格）。

```
char story[100];
cout << "写一句话:";
cin.getline(story, 100);   // 读取整行
cout << "你写了:" << story;
```

考试注意事项：

易错点	正确写法说明
忘记加 #include <cstring>	所有 str 函数都需要这个头文件
strcpy(a, b) 写成 a = b	字符数组不能直接赋值，要用函数复制
下标超出范围	定义数组长度要足够（含 \0 终止符）

（二）string 类与相关函数（string Class & Functions）

☑什么是 string 类?

string 是 C++ 提供的更强大、更好用的字符串类型，使用方式更简单、更安全，不需要手动处理 '\0'。

需要引入头文件：**#include <string>**

示例代码：

```
#include <iostream>
#include <string>
using namespace std;
int main() {
```

```
    string name = "Alice";
    cout << name << endl;
    return 0;
}
```

☑ **常用 string 函数和操作：**

1. length() / size()

作用：测量字符串长度（字符个数），像量身高一样。

```
string fruit = "apple";
cout << fruit.length();  // 输出 5
```

2. empty()

作用：检查字符串是否为空（长度为 0），是则返回 true，否则返回 false。（像检查书包）

```
string bag = "";
if (bag.empty()) {
    cout << "书包是空的！";  // 会输出这句
}
```

3. find()

作用：在大字符串中查找小字符串或字符第一次出现的位置。如果找到，返回位置（下标）；没找到，返回 string::npos（一个特殊值，通常为-1）（像玩藏宝图）

```
string map = "宝藏藏在森林里";
int pos = map.find("森林");  // pos=9（找到的位置）
if (pos == string::npos) {
    cout << "没找到宝藏！";  // npos 表示没找到
}
```

4. substr()

作用：截取字符串的一部分（像切蛋糕）

```
string cake = "巧克力草莓蛋糕";
string piece = cake.substr(3, 4); // 从位置 3 切 4 个字符
cout << piece;  // 输出 "草莓"
```

5. replace()

作用：替换字符串的一部分（像修改错字）

```
string msg = "我讨厌数学";
msg.replace(1, 2, "喜欢"); // 从位置 1 开始替换 2 个字
cout << msg;  // 输出 "我喜欢数学"
```

6. insert()

作用：在字符串中插入字符（像插队）

```
string line = "排队：小红小明";
line.insert(7, "小刚、");  // 在位置 7 插入
cout << line;  // 输出 "排队：小红、小刚小明"
```

7. erase()

作用：删除一部分字符串（像用橡皮擦）

```
string str = "ABCD1234EFGH";
str.erase(4, 4);  // 从位置 4 删除 4 个字符
cout << str;  // 输出 "ABCDEFGH"
```

8. append() 或+= 操作符

作用：在字符串末尾添加另一个字符串（或字符）。（像拼积木）

示例 1：

```
string a = "Hello";
string b = "World";
a += " " + b;  // 拼接字符串
cout << a;  // 输出 "Hello World"
```

示例 2：

```
string s1 = "Hello";
string s2 = " World";
s1.append(s2); // s1 变为 "Hello World"
s1 += "!";       // s1 变为 "Hello World!"
```

9. compare()

作用：比较两个字符串的大小（字典序）。如果相等返回 0；如果调用对象小于参数，返回负数；大于则返回正数。（像比身高）

```
string s1 = "apple";
string s2 = "banana";
if (s1.compare(s2) < 0) {
    cout << "apple 在字典里更靠前";  // 会输出
}
```

10. getline()

作用：读取整行输入（包含空格）

```
string story;
cout << "输入一句话：";
getline(cin, story);  // 读取整行
```

```
// 输入："C++很有趣"
cout << story;  // 完整输出
```

1. front() 和 back()

作用：front()返回字符串的第一个字符，back()返回最后一个字符。

```
string s = "Hello";
cout << s.front(); // 输出 'H'
cout << s.back();  // 输出 'o'
```

2. stoi(), stol(), stoll(), stof(), stod() 等

作用：将字符串转换为整数、长整数、浮点数等。

```
string s1 = "123";
int n = stoi(s1); // n=123
string s2 = "3.14";
double d = stod(s2); // d=3.14
```

3. to_string()

作用：将数值（整数、浮点数等）转换为字符串。

```
int n = 123;
string s = to_string(n); // s="123"
double d = 3.14;
string s2 = to_string(d); // s2="3.140000"（注意可能有多余的 0）
```

总结：char[] 和 string 区别对比

比较项	字符数组 char[]	string 类
是否要写结束符	需要 \0	不需要 \0

易用性	较复杂，需手动处理	☑ 更简单
功能支持	需 cstring 函数辅助	有成员函数，支持直接操作
安全性	容易越界	较安全
举例	char name[10] = "Hi";	string name = "Hi";