笔者注:

本文系个人整理，已进行原创声明，如需转载或发表请联系作者。

联系方式 :

- 公众号 -- python入门到进阶
- 微　信 -- W656824258

勘误:

由于我自己也不是资深编程高手，在创作此内容时尽管已经力求精准，查阅了诸多资料，还是难保有所疏漏，如果各位发现有误可以微信联系，欢迎指正。

4、结构体数组
5、运算符重载
6、结构体函数
7、结构体指针

C++ 高级概念

1、创建类
2、构造函数
3、...

常见问题解决方案

1、反转数组

C++ 基本概念

一、第一个C++程序

```
#include <iostream>
using namespace std;

int main(){
    cout << "Hello World";
    return 0;
}
```

二、基本输入输出语句

```
// 声明一个变量
int number = 1;

// 声明一个常量
const double pi = 3.14;

// 数学表达式
int x = 10 + 3;

// 在控制台输出内容
cout << "x=" << x  << endl;  // endl 表示换行

// 从控制台读入数据
cin >> number;
```

三、数据类型

1、一些常用数据类型

```
// 数据类型
int age = 18;                   // 整型
double price = 9.99;            // 双精度浮点数
float interestRate = 3.67F;     // 单精度浮点数
long fileSize = 90000L;         // 长整型
char letter = 'a';              // 字符型
string name = "Wzs";            // 字符串
bool isValid = true;            // 布尔类型
auto years = 5;                 // 自动识别数据类型 (C++11及以后支持)
```

2、数字系统

```
int x = 255;                    // 十进制数
int y = 0b111111;               // 二进制数
int z = 0xFF;                   // 十六进制数
```

3、数据类型占据空间和范围

```
int bytes = sizeof(int);
int min = numeric_limits<int>::min();  // int 类型所能存储最小值
int max = numeric_limits<int>::max();  // int 类型所能存储最大值
```

4、数组

```
int numbers[] = {1, 2, 3};
cout << numbers[0];           // 理解数组下标从零开始即可
```

5、数据类型转换

```
// C语言风格的数据类型转换
double a = 2.0;
int b = (int) a;

// C++风格的数据类型转换
int c = static_cast<int>(a);
```

四、判断语句

1、比较运算符

```
bool a = 10 > 5;
bool b = 10 == 10;
bool c = 10 != 5;
```

2、逻辑运算符

```
bool d = a && b;                // 逻辑与   --->  左右同时为真，结果为真
bool e = a || b;                // 逻辑或   --->  左右一边为真，结果为真
bool f = !a;                    // 逻辑非   --->  真变假，假变真
```

3、if....else if...else语句

```
if (temperature < 60){
  // ...
}
else if (temperature < 90){
  // ...
}
else{
  // ...
}
```

4、条件运算符

```
double commission = (sales < 10'000) ? .05 : .1;
```

5、switch语句

```
switch (menu){
  case 1:
    // ...
    break;
  case 2:
    // ...
    break;
  // ...
  default:
    // ...
}
```

五、循环语句

1、for循环

```
for (int i = 0; i < 5; i++)
    cout << i;
```

2、while循环

```
int i = 0;
while (i < 5){
  cout << i << endl;
  i++;
}
```

3、do...while循环

```
int i = 0;
do{
  cout << i << endl;
  i++;
}while (i < 5);
```

4、基于范围的for循环

```
int numbers[] = {1, 2, 3};
for (int number: numbers)
    cout << number << endl;
```

六、函数

1、定义函数

```
// 无返回值的函数
void greet(string name){
    cout << "Hello " << name;
}

// 有返回值的函数
string fullName(string firstName, string lastName){
    return firstName + " " + lastName;
}
```

2、使用默认值参数

```
double calculateTax(double income, double taxRate = .2){
    return income * taxRate;
}
```

3、函数重载

```
void greet(string name){
    cout << "Hello " << name;
}

void greet(string title, string name){
    cout << "Hello " << title << " " << name;
}
```

4、引用传参

```
void incresePrice(double &price){
    price *= 1.2;
}
```

5、函数声明

```
void greet(string name);
```

6、定义命名空间

```
namespace messaging{
    void greet(string name){}
}
```

7、使用命名空间

```
using namespace messaging
// or
using messaging::greet;
```

C++ 中级概念

一、数组

1、创建并初始化数组

```
int numbers[] = {1, 2, 3};
string names[5];
```

2、访问数组元素

```
numbers[0] = 10;
cout << numbers[0];
```

3、确定数组的大小

```
auto size = size(numbers);
```

4、数组解包

```
auto [x, y, z] = numbers;      // 新特性
```

5、二维数组

```
int matrix[2][3] = {
    {11, 12, 13},
    {21, 22, 23}
};
matrix[0][0] = 10;
```

二、指针

1、声明和使用指针

```
int number = 10;
int* ptr = &number;
*ptr = 10;
```

2、指向常量数据的指针

```
const int x = 10;
const int* ptr = &x;
```

3、常量指针

```
int x = 10;
int* const ptr = &x;
```

4、指向常量数据的常量指针

```
int x = 10;
const int* const ptr = &x;
```

5、使用原始指针动态内存分配

```
int* numbers = new int[10];
delete[] numbers;
```

6、使用智能指针动态内存分配

```
#include <memory>
auto numbers:unique_ptr<int[]> = make_unique<int[]>(10);
```

三、字符串

1、使用C字符串

```
char name[5] = "Mosh";
char copy[5];

cout << strlen(name);            // 求解字符串长度

strcpy(copy, name);              // 拷贝字符串

if (strcmp(name, copy) == 0){    // 判断两个字符串是否相等
    cout << "Equal";
}
```

2、使用C++字符串

```
string name = "python入门到进阶";

cout << name.length();           // 求解字符串长度

string copy = name;              // 拷贝字符串

if (name == copy)                // 判断字符串是否相等
    cout << "Equal";
```

3、修改字符串

```
string name = "Zesheng";
name.append(" Wang");            // 拼接字符串
name.insert(0, "I am ");         // 在索引位置插入字符串
name.erase(0, 2);                // 删除索引从 0 到 1 的字符
name.clear();                    // 删除字符串中的所有字符
name.replace(0, 2, "**");        // 将索引 0 到 1 的字符（"Ze"）替换成 "**"
```

4、查找字符串

```
string name = "Wzs";
int index;
index = name.find('a');          // 查找指定字符位置 可以传递两个参数，待搜索字符和起始搜索
位置
                                 // 查找失败会返回-1

index = name.rfind('a');         // 从右向左搜索字符 'a' 最后一次出现的位置。
                                 // 如果没有找到字符 'a'，则返回一个特殊值 string::npos

// 下面的方法是一个用于在字符串 name 中查找第一个出现在指定字符集合 ",.;" 中的任意字符的函数。
index = name.find_first_of(",.;");

// 下面的方法是一个用于在字符串 name 中查找最后一个出现在指定字符集合 ",.;" 中的任意字符的函
数。
index = name.find_last_of(",.;");
```

```
//  下面的方法是一个用于在字符串 name 中查找第一个不属于指定字符集合 ",.;" 中的字符的函数。
index = name.find_first_not_of(",.;");
```

5、提取子串

```
string name = "Zesheng Wang";
string substr;

substr = name.substr();         // 不提供参数则拷贝字符串
substr = name.substr(3);        // 是一个用于从字符串 name 中提取子字符串的函数。
substr = name.substr(3, 5);     // 从索引 3 开始的 2 个字符构成了子字符串
```

6、尝试使用字符

```
string name = "Zesheng Wang";
bool b;
b = isupper(name[0]);           // 判断是否是大写字符
b = islower(name[0]);           // 判断是否是小写字符
b = isdigit(name[0]);           // 判断是否是数字字符
b = isalpha(name[0]);           // 判断是否是字母字符

name[0] = toupper(name[0]);     // 转换为大写字母
name[0] = tolower(name[0]);     // 转换为小写字母
```

7、字符串转换

```
string str = "10";
int i = stoi(str);              // 字符串转换为整数
double d = stod(str);           // 字符串转换为浮点数
string s = to_string(10);       // 整数转换为字符串
```

8、转义字符

```
// 换行符
string message = "Hello\nWorld";

// 制表符
string columns = "first\tlast";

// 输出反斜杠
string path = "c:\\folder\\file.txt";
```

9、原始字符串

```
string path = R"(c:\folder\file.txt)";
```

四、结构体

1、定义结构体

```
struct Movie
{
    string title = " ";
    int releaseYear = 0;
};
```

2、创建一个结构体实例

```
Movie movie = {"灌篮高手", 2023};
```

3、结构解包

```
auto [t, r]{movie};  // c++17之后才能使用的语法
```

4、结构体数组

```
struct Movie
{
    string title;
    int releaseYear = 0;
    bool isPopular;
};

int main()
{
    vector<Movie> movies;
    Movie movie{"terminator", 1984};
    movies.push_back(movie);
    movies.push_back(movie);

    cout << movies[0].title << endl;
    for (Movie movie : movies)
    {
        cout << movie.title << endl;
    }

    return 0;
}
```

5、运算符重载

```
bool operator==(const Movie &first, const Movie &second)
{
    return (first.title == second.title &&
            first.releaseDate.year == second.releaseDate.year;
}
```

6、结构体函数

```
#include <iostream>
#include <iomanip>
#include <cstring>
#include <vector>

using namespace std;
struct Date {
    short year = 1900;
    short month = 1;
    short day = 1;
};

struct Movie {
    string title;
    Date releaseDate;
    bool isPopular;


};

bool operator==(const Movie &first, const Movie &second) {
    return (first.title == second.title &&
            first.releaseDate.year == second.releaseDate.year &&
            first.releaseDate.month == second.releaseDate.month &&
            first.releaseDate.day == second.releaseDate.day);
}

ostream& operator<<(ostream& stream, const Movie& movie){
    stream << movie.title;
    return stream;
}

Movie getMovie(){
    return {"terminator", 1984};
}

void showMovie(Movie& movie){
    cout << movie.title;
}
int main() {
    auto movie = getMovie();
    showMovie(movie);
    return 0;
}
```

7、结构体指针

```
void showMovie(Movie* movie)
{
    cout << movie->title;
}
```

C++ 高级概念

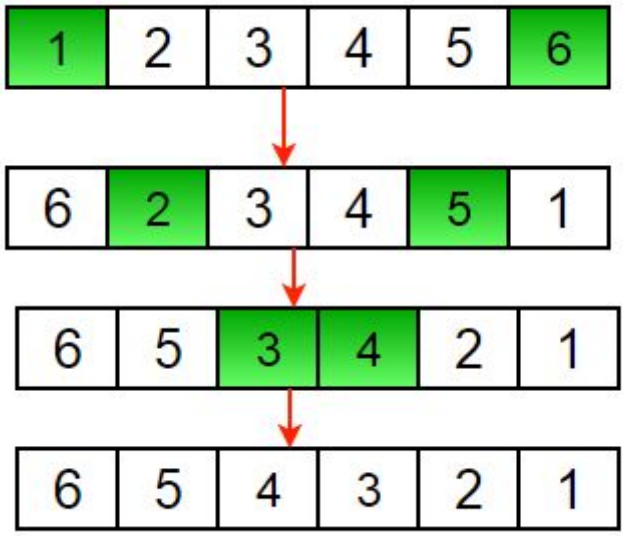
1、创建类

2、构造函数

3、...

常见问题解决方案

1、反转数组




```
// 反转数组
#include <bits/stdc++.h>
using namespace std;

/* Function to reverse arr[] from start to end*/
void rvereseArray(int arr[], int start, int end)
{
    while (start < end)
    {
        int temp = arr[start];
        arr[start] = arr[end];
        arr[end] = temp;
        start++;
        end--;
    }
}

/* 打印一个数组 */
void printArray(int arr[], int size)
{
    for (int i = 0; i < size; i++)
        cout << arr[i] << " ";
    cout << endl;
}

int main()
{
    int arr[] = {1, 2, 3, 4, 5, 6};
    int n = sizeof(arr) / sizeof(arr[0]);
    printArray(arr, n);
    // 函数调用
    rvereseArray(arr, 0, n-1);
    cout << "Reversed array is" << endl;
    printArray(arr, n);
```

```
    return 0;
}
```