lower\_bound(int k)二分查找第一个不小于k的下标，找不到返回end

upper\_bound(int k)二分查找第一个大于k的下标

对于集合lower\_bound是内置函数，set.lower\_bound(int k)返回迭代器，找不到返回end

对于vector，auto it = lower\_bound(v.begin(), v.end(), 4);lower\_bound在algorithm中定义

求最大公约数

int std::gcd(int a, int b);c++17可用

复制粘贴神器：cout<<R”(复制内容)”；c++11引入

可以自定义struct比较器，重写operator函数，实现自定义的map等

struct CompareByLength {

bool operator()(const std::string& a, const std::string& b) const {

return a.length() < b.length(); // 按照字符串长度升序排序

}

};

 **const 放在函数尾部**：保证该成员函数不会修改类的成员变量，因此这是一个常量成员函数。

 **const 修饰参数**：确保传入的参数是常量引用，这避免了不必要的复制，并且保证函数内不会修改这些参数。

map<std::string, int, CompareByLength> myMap;

set<std::string, CompareByLength> mySet;

priority\_queue<Person, std::vector<Person>, CompareByAge> pq;

[捕获方式](参数列表) -> 返回类型 {

    函数体

};

**按值捕获（[=]）**：

* 通过值的方式捕获外部变量，lambda 表达式内部获得外部变量的拷贝。

**按引用捕获（[&]）**：

* 通过引用的方式捕获外部变量，lambda 可以直接修改外部变量的值。

**显式捕获特定变量**：

* 可以显式指定某个变量按值或按引用捕获。
* int x = 5, y = 10;
* auto func = [x, &y]()
* {
* // x 是按值捕获，不能修改；y 是按引用捕获，可以修改
* cout << x << " " << y << endl;
* y = 20;
* };
* func();            // 输出 5 10
* cout << y << endl; // 输出 20

**捕获 this 指针（[this]）**：

* 在类的成员函数中，lambda 表达式可以捕获 this 指针，以便访问成员变量。

#define int long long

signed main()