多线程编程是一种并发编程的方式，它允许程序同时执行多个线程，以提高程序的性能和效率。以下是多线程编程的基础知识以及相关的C++标准库类和概念：

线程（Thread）：线程是程序中独立执行的最小单位，它包含了代码的执行上下文和共享的资源。一个程序可以同时创建多个线程，并且这些线程可以并行执行。

并发（Concurrency）：并发指的是多个任务在同一时间段内交替执行的状态。通过多线程编程，可以实现并发执行，提高程序的响应性和吞吐量。

互斥（Mutex）：互斥是一种线程同步机制，用于保护共享资源的访问。只有持有互斥锁的线程可以访问被保护的资源，其他线程需要等待锁的释放。

std::thread：std::thread是C++11标准库提供的用于创建和管理线程的类。它可以接受一个可调用对象作为线程的入口点函数，并提供了线程的创建、启动、等待等功能。

std::mutex：std::mutex是C++11标准库提供的互斥锁类，用于保护共享资源的访问。它提供了互斥锁的加锁和解锁操作，并可以通过递归锁避免死锁情况。

std::lock\_guard：std::lock\_guard是C++11标准库提供的互斥锁保护类，它可以自动管理互斥锁的生命周期。在创建std::lock\_guard对象时加锁，在对象析构时自动解锁，避免了手动管理锁的问题。

std::condition\_variable：std::condition\_variable是C++11标准库提供的条件变量类，用于线程间的同步和通信。它可以配合互斥锁一起使用，实现线程的等待和唤醒操作。

线程池（Thread Pool）：线程池是一种管理和复用线程的机制，它可以避免频繁创建和销毁线程的开销。线程池维护一定数量的线程，并将任务分配给空闲的线程执行。