**Race conditions**

Critical section is any piece of code that has the possibility of being executed concurrently by more than one thread of the application and exposes any shared data or resources used by the application for access.

Race conditions happen when threads run through critical sections without thread synchronization. The threads “race” through the critical section to write or read shared resources and depending on the order in which threads finish the “race”, the program output changes.

In a race condition, threads access shared resources or program variables that might be worked on by other threads at the same time causing the application data to be inconsistent.

# How to avoid race conditions?

Race conditions occur within the critical section of your code. These can be avoided with proper thread synchronization within critical sections by using techniques like locks, atomic variables, and message passing.