1.Средства работы с многопоточностью в стандартной библиотеке появились в стандарте языка Java 1.2, который был выпущен в декабре 1998 года. Этот стандарт добавил пакет java.util.concurrent, который содержит классы и интерфейсы для работы с потоками исполнения и синхронизации доступа к общим ресурсам в многопоточных приложениях. С тех пор этот пакет был расширен и улучшен в последующих версиях языка Java.

2. Состояние гонки (race condition) - это ситуация в многопоточном программировании, когда два или более потока пытаются одновременно обратиться к одному и тому же ресурсу (например, переменной), и результат зависит от того, какой поток выполнится первым. Это может привести к непредсказуемому поведению программы, ошибкам в работе и даже краху системы.

Например, предположим, что два потока хотят увеличить значение переменной на 1. Если оба потока выполнят операцию чтения и записи одновременно, то может произойти так, что оба потока прочитают одно и то же значение, увеличат его и запишут обратно, причем одно из изменений будет перезаписано другим. В результате переменная будет увеличена только на 1, вместо 2, что было намерением каждого из потоков.

Для предотвращения состояний гонки многопоточные программы должны использовать механизмы синхронизации, такие как блокировки, семафоры, мьютексы, атомарные операции и другие, которые обеспечивают правильный доступ к общим ресурсам в разных потоках.

3. Мьютекс (mutex) - это механизм синхронизации, который позволяет предотвратить состояние гонки и обеспечить правильный доступ к общим ресурсам в многопоточных программах.

Мьютексы используются в тех случаях, когда необходимо синхронизировать доступ к ресурсу между двумя или более потоками. Мьютексы позволяют защитить критические секции кода, т.е. участки кода, которые могут одновременно выполняться только одним потоком, чтобы гарантировать правильность выполнения операций в этой критической секции.

Когда один поток получает доступ к мьютексу, он блокирует другие потоки от доступа к критической секции кода. После завершения работы с общим ресурсом первый поток освобождает мьютекс, и другие потоки могут получить доступ к ресурсу.

Мьютексы широко используются в программировании для синхронизации доступа к разделяемым данным, критическим секциям и ресурсам. Они обеспечивают безопасный и согласованный доступ к общим ресурсам в многопоточных приложениях.

3. Устройства хранения информации в компьютере можно расположить по скорости доступа к данным от самого быстрого до самого медленного следующим образом:

Оперативная память (RAM) - это тип памяти, который используется для временного хранения данных и программ, которые в настоящее время используются компьютером. Оперативная память имеет очень быстрый доступ к данным, поэтому это самое быстрое устройство хранения информации в компьютере.

Кэш-память (Cache) - это тип памяти, которая используется для ускорения доступа к данным, которые часто используются процессором. Кэш-память расположена на процессоре, что позволяет сократить время доступа к данным до нескольких тактов процессора.

SSD-накопители (Solid State Drives) - это тип накопителей, который использует флэш-память для хранения данных. SSD-накопители имеют быстрый доступ к данным, но медленнее, чем оперативная память и кэш-память.

Жесткие диски (Hard Disk Drives) - это тип накопителей, который использует магнитную запись для хранения данных на вращающихся магнитных дисках. Жесткие диски имеют медленный доступ к данным, поэтому они являются наиболее медленным устройством хранения информации в компьютере.

Значения скорости доступа к данным устройств хранения информации могут отличаться в зависимости от конкретных моделей и производителей.

4.Закон Амдала утверждает, что при увеличении числа процессоров в вычислительной системе с фиксированным размером задачи, ускорение вычислений ограничено долей вычислений, которые нельзя распараллелить (параллельная доля).

Формула закона Амдала имеет вид:

S(n) = 1 / [ (1-P) + P/n ],

где S(n) - ускорение вычислений при использовании n процессоров, P - доля вычислений, которые можно распараллелить.

Таким образом, закон Амдала указывает на ограничения, связанные с распараллеливанием задач. Для увеличения скорости выполнения задачи необходимо увеличивать долю параллельных вычислений. При этом при достижении определенного количества процессоров, дальнейшее увеличение их числа может не дать значительного прироста производительности из-за ограничений параллельной доли.

Однако, закон Амдала является упрощенной моделью и не учитывает многие факторы, такие как накладные расходы на коммуникацию между процессорами, время доступа к общей памяти и др. Поэтому реальное ускорение вычислений может отличаться от значения, которое было бы предсказано формулой закона Амдала.