Week 2

**What causes false sharing?**

Unterschiedliche Threads von verschiedenen Prozessoren schreiben (ändern) Variablen, die innerhalb der gleichen cache-line gespeichert sind.

**How do mutual exclusion constructs prevent race conditions?**

Durch ein exclusion construct wird sichergestellt, dass zu einer gegebenen Zeit jeweils nur ein thread auf eine Variable zugreifen kann. Dadurch wird gleichzeitiges schreiben auf die selbe Variable verhindert.

**Explain the differences between static and dynamic schedules in OpenMP.**

Static scheduling verteilt Arbeitspakete gleichmäßig auf verfügbare Threads, während beim dynamischen scheduling ein Thread ein neues Arbeitspaket bekommt sobald es mit dem vorigen fertig ist. D.h. bei Arbeitspaketen mit unterschiedlichem Arbeitsaufwand sollte man dynamisches scheduling verwenden; wenn jedoch alle Arbeitspakete den gleichen Aufwand benötigen, benutzt man static scheduling, um effizienter zu bleiben (static scheduling ist effizienter als dynamic scheduling, da keine extra Entscheidung getroffen werden muss, welches Paket von welchem Thread behandelt wird).

Die Entscheidung über das scheduling wird bei static – zur compile Zeit getroffen.

bei dynamic – zur Laufzeit getroffen.

**What can we do if we've found a solution while running a parallel for loop in OpenMP, but still have many iterations left?**

Use a boolean variable to check whether we have found our solution yet.

If that variable is set to true, we can just use 'continue' to keep iterating through the for loop without doing any more work.

**Do the coding warmup on slide 22. Explain in your own words how std::atomic::compare_exchange_weak work.**