Task 2:

1. private or shared in omp parallel:
   1. i : private
   2. j : shared
   3. g1 : private
   4. g2 : shared
2. private or shared in foo:
   1. p : private
   2. g1 : shared
   3. g2 : shared

Task 6:

Durch private bekommt jeder Thread seine eigene private variable a die jedoch **nicht initialisiert** (firstprivate übernimmt zuvor beinhaltete Werte mit in den parallelen Aufruf) ist. Das führt somit zu einem Fehler beim inkrementierten.

Allerdings wurde auf manchen unserer Geräte a automatisch mit 0 initialisiert, wodurch jeder Thread 1 ausgab. Dies geschah in Anlehnung an das oben genannte jedoch immer, unabhängig davon, wie a vor dem parallelen Aufruf initialisiert wurde. Ob ein Fehler ausgegeben wurde oder a automatisch mit 0 initialisiert hängt vermutlich an unterschiedlichen Versionen zusammen.