Semestrální práce – KIV/PGS

## Struktura programu

### Main

Načte vstupní parametry z příkazové řádky a vypíše je. Vytvoří instance ostatních tříd potřebných k běhu programu. Spustí simulaci.

### Simulation

Celá simulace je reprezentována v této třídě. Má jako atributy uloženy instance všech ostatních tříd potřebných k simulaci a tyto třídy pak komunikují a volají své metody přes Simulation. Jediná třída, která není volána přes simulation je Barrier, která je uložená jako atribut třídy Ferry, která volá její metodu.

## Kritické sekce

### Získávání bloku pro zpracování

Získávání bloku od foremana ze zpracování je první kritickou sekcí, protože by se mohlo stát, že 2 workeři požádají o blok současně a dostanou oba stejný blok. Aby k tomu nedošlo, je metoda getBlock() ve třídě simulation, synchzonized. Až metodu nějaký worker zavolá tak se zamkne a dokud nedostane blok, tak metodu jiný worker volat nemůže.

### Nakládaní zdrojů na lorry

Další kritickou sekcí programu je nakládání zdrojů na lory, protože velmi často bude metodu pro naložení volat worker ve chvíli, kdy ještě nakládá jiný. Proto musí být tato metoda také synchronized.

### Ferry čeká na naplnění

V momentě, kdy se naplní lorry je třeba jeho vlákno pozastavit, dokud se nenaplní kapacita ferry. Tato kritická sekce je ošetřena pomocí bariéry. Ta je reprezentována vlastní třídou Barrier, která má jen jednu synchronized metodu synchronize(), kterou volá ferry po naložení každé lorry. Metoda zajistí, aby vlákna lorries byla uspána, dokud není kapacita ferry naplněna. Bariéra je navíc ošetřena proti samovolnému vzbuzení vláken pomocí booleanu sleep, který se nastaví na false a nepustí do bariéry další lorries, dokud na bariéře ještě čeká lorry z předchozího ferry.

### Ukončení simulace

Poslední kritickou sekcí programu je ukončení simulace. Aby se nestalo, že se vypíší výsledky, poté co dělníci vytěží všechny zdroje, ale náklaďáky nejsou v cíli, je z metody *main()*, volána metoda *waitUntilOver()*. Tato metoda volá nad všemi vlákny dělníků a náklaďáků *join()*, aby simulace počkala než skončí. Pro jistotu je zde ještě cyklus který volá nad objektem *changedExtracted* příkaz *wait()*, dokud není roven počet vytěžených a počet nalezených zdrojů. Nad objektem *changedExtracted* je voláno notify(), pokaždé, když se navyšuje počet zdrojů dovezených do cíle.