# 3. domácí úloha: Binární vyhledávací strom

Binární vyhledávací stromy patří mezi základní algorimické datové struktury (setkali jste se s nimi například na kurzu Algoritmizace). Vaší úlohou v této domácí úloze bude takový binární vyhledávací strom naimplementovat do souborů `bst_tree.h` a `bst_tree.cpp`, které najdete v tomto balíčku [https://cw.fel.cvut.cz/wiki/_media/courses/b4b36pdv/tutorials/hw04_cds.zip]. Narozdíl od algoritmizace po Vás budeme chtít verzi binárního stromu, ke které bude moci více vláken přistupovat současně a nebudou na sebe muset zbytečně čekat (pokud to opravdu nebude nutné). Jelikož je konkurentní verze složitější, nemusíte řešit vyvažování stromu a bude nám stačit, pokud naimplementujete metodu `insert`.

**Zadání:** Do souboru `bst_tree.cpp` doimplementujte tělo metody `insert`. Tato metoda vloží nový uzel do datové struktury binárního stromu na odpovídající místo (tj., aby bylo dodržené uspořádání na uzlech). Soubory `bst_tree.cpp` a `bst_tree.h` si můžete upravovat dle Vaší potřeby (například si můžete doplnit vlastní metody nebo si přidat členské proměnné tříd). Zachovejte ale prosím následující:

1. Uzly reprezentujte pomocí typu `bst_tree::node`.
2. Pointer na kořen stromu ponechte v členské proměnné `root` třídy `bst_tree`.
3. Pointer na levý podstrom aktuálního uzlu ukládejte do členské proměnné `left` třídy `bst_tree::node`.
4. Pointer na pravý podstrom aktuálního uzlu ukládejte do členské proměnné `right` třídy `bst_tree::node`.
5. To, že daný podstrom neexistuje, indikujte tím, že daný pointer má hodnotu `nullptr`.

Tyto vlastnosti využíváme pro kontrolu správnosti Vašeho řešení (kontrolní kód si můžete prohlédnout v souboru `tests.h`). Můžete si ale pointery změnit na atomické (tj., typ `std::atomic<node*>`).

Při implementaci konkurentních datových struktur je důležitý princip optimistického zamykání, abyste mohli dosáhnout maximálního paralelismu. Naše referenční implementace je implementovaná bez zámků za pomocí atomických operací. Pokud se přiblížíte času referenční implementace, získáte **2 body**. V případě, že Vaše řešení bude pomalejší, získáte bodů méně (body jsou odstupňované po půl bodech).

**Odevzdání:** Do BRUTE odevzdávejte archiv obsahující Vaši implementaci souborů `bst_tree.cpp` a `bst_tree.h`. Před odevzdáním se ujistěte, že jste z kódu odebrali všechny ladící výpisy. Termíny odevzdání jsou 23.3. 23:59 (platí pro studenty zapsané na středečních cvičeních) a 24.3. 23:59 (platí pro studenty zapsané na čtvrtečních cvičeních).