

zaawansowane techniki C++ i STL

metaprogramowanie

Instytut Informatyki
Uniwersytetu Wrocławskiego

Paweł Rzechonek

Zadanie 1 (1 pkt)

Wykorzystując metaprogramowanie zdefiniuj szablon obiektu funkcyjnego, pozwalającego obliczyć na poziomie kompilacji największy wspólny dzielnik $\text{NWD}(a, b)$ dla liczb naturalnych $a, b \geq 1$ dla naturalnych liczb $0 \leq k \leq n$. Funkcja powinna działać (w trakcie kompilacji) w logarytmicznym czasie $O(\log a + \log b)$.

Zadanie 2 (2 pkt)

Wykorzystując metaprogramowanie zdefiniuj szablon obiektu funkcyjnego, pozwalającego obliczyć na poziomie kompilacji współczynnik dwumianowy $\binom{n}{k}$ dla liczb naturalnych $0 \leq k \leq n$. Funkcja powinna działać (w trakcie kompilacji) w liniowym czasie $O(n)$.

Zadanie 3 (2 pkt)

Wykorzystując metaprogramowanie zdefiniuj szablon obiektu funkcyjnego, pozwalającego obliczyć na poziomie kompilacji n -tą liczbę Lucasa dla liczby naturalnej $n \geq 0$. Funkcja powinna działać (w trakcie kompilacji) w liniowym czasie $O(n)$.

Zadanie 4 (2 pkt)

Wykorzystując metaprogramowanie zdefiniuj szablon funkcji liczącej iloczyn skalarny dwóch wektorów:

```
template<size_t N, typename T = double> T inner(T *x, T *y);
```

Pierwszym parametrem tego szablonu ma być długość mnożonych wektorów (tablice z wartościami) a drugim typ wartości (domyślnie `double`).

Przetestuj działanie funkcji szablonowej `inner()` na danych typu `int` oraz `complex<double>`.

Zadanie 5 (3 pkt)

Wykorzystując metaprogramowanie zdefiniuj szablon klasy będącej opakowaniem dla obiektu określonego w parametrze typu `template<typename T> struct obj_holder`. Opakowanie to ma konieczne zaalokować pamięć dla obiektu na stosie, jeśli jest on mały (jego rozmiar jest nie większy niż `sizeof(string)`) i nie jest tablicą albo na sterpie w przeciwnym przypadku.

W strukturze `obj_holder` zdefiniuj alias `type` dla docelowego typu ustalony w oparciu o selektor `std::conditional<>` – będzie to albo struktura `on_stack` (dla małych obiektów) albo `on_heap` (dla dużych obiektów tworzonych dynamicznie na sterpie, pamiętaj aby w destruktorze tego opakowania zwolnić przydzieloną pamięć). W obu tych strukturach zaimplementuj semantykę

wskaźników, czyli operator wyłuskania (gwiazdka *) udostępniający referencję do obiektu i operator dostępu do składowych wewnątrz struktur/klas (strzałka ->) udostępniający adres obiektu.