Zaawansowane programowanie C++

Lista 2 - Szablony część 2

Zadanie 5

Skopiuj plik z rozwiązaniem zadnia 1 z listy 1 i dopisz w nim:

- specjalizację szablonu funkcji *add1* dodającą dwie dowolne liczby przekazane przez wskaźniki,
- wersję funkcji *add1*, która będzie łączyć ze sobą dwa teksty przekazane jako *const char**.

Do funkcji main dodaj wywołania sprawdzające poprawność działania dopisanych funkcji.

Zadanie 6

Metaprogramowanie

Napisz szablon, który będzie mógł być użyty do obliczenia w czasie kompilacji objętości hipersześcianu o dowolnej, całkowitej długości boku i ilości wymiarów.

W przypadku podania ujemnej wartości długości boku lub ilości wymiarów, powinna zostać zwrócona wartość -1.

Zadanie 7

If constexpr

Napisz szablon funkcji *add*, który będzie dodawać dowolną liczbę obiektów zasadniczo dowolnych typów, np. *add(1, 1.0, 1.0f);*
Ma to być jeden szablon.
Wskazówka: zaprzyjaźnij się z operatorem sizeof…

Problemy do dyskusji w grupie

Proszę przedyskutować w grupie poniższy kod, każdy wiersz, każdą instrukcję, tak, by go dokładnie zrozumieć. Następnie proszę sprawdzić, czy kompilator w rybie *Release* rozwija wyrażenie *table<int, 10>[6]* do wartości 36.

Wskazówki:
- można posłużyć się kompilatorem online, https://godbolt.org/ - wybierz w nim możliwie najnowszy kompilator gcc lub clang, ustaw C++ w wersji co najmniej 17 (-std=c++17), tryb Release (-O3) lub bez optymalizacji (-O0).
- W komentarzach do kodu wskazuję miejsca, na które warto zwrócić uwagę.
  Jeśli ktoś wie, co w żargonie programistów C++ oznacza NRVO, to może wskazać miejsce, gdzie ta optymalizacja jest zapewne używana.

```
1.  #include <array>
2.  #include <iostream>
3.
4.  template<typename T, T SIZE>          // T SIZE?
5.  constexpr auto table = []{            // constexpr? auto? []? brak ()?
6.    std::array<T, SIZE> result = {};   // std::array? = {}?
7.    for (T i = 0; i < SIZE; ++i)
8.    {
9.      result[i] = i * i;
10.   }
11.   return result;                     // typ wyniku?
12. }();                                 // ()? ;?
13.
14. int main()
15. {
16.   std::cout << table<int, 10>[6]     // typ table? [6]?
17.             << std::endl;
18. }
```