**Задача №1**

Реализуйте шаблонный класс shared\_ptr.

**Задача №2**

Напишите шаблонную функцию

    bool close\_enough(T a, T b)

,которая сравнивает два числа. Если числа не с плавающей точкой, то функция работает как обычный оператор сравнения (==), иначе функция сравнивает разность между числами и некоторой малой величиной.

**Задача №3**

Написать вариадическую функцию со следующей сигнатурой

template< class ... Args >

auto sum( Args ... values )

которая суммирует свои аргументы

sum( 1, 2, 3 ) => 1 + 2 + 3

**Реализовать два варианта программы**:

1. С разворачиванием аргументов через вспомогательные функции

auto sumI( T value )

auto sumI( T value, Args ... other\_values )

2. С использованием fold выражения ( val + ... )

**Задача №4**

Обобщить код из задачи 3 на любые бинарные операции задаваемые через лямбды или с помощью вспомогательных классов std::multiplies, std::plus и т.п..

Переделать функцию

template< class ... Args >

auto sum( Args ... values )

в функцию

template< class Op, class ... Args >

auto reduce( Op f, Args ... values )

В которой Op f это бинарная операция представляемая лямбдой или любым классом в котором определена операция operator()

**Задача №5**

Реализовать функцию принимающую две функции с одинаковыми аргументами и возвращающую функцию, которая будет возвращать результат произведения двух исходных функций. Для определения на стадии компиляции, какие аргументы принимаются функцией, реализовать вспомогательный класс type traits.

template<class A, class B>

auto multiply\_functions( A f1, B f2 )

f1(x,y) → x + y

f2(x,y) → x / y

multiply\_functions( f1, f2 )

|

|

V

f3(x,y) → x^2 / y + x

**Задача №6**

**Дан класс**

**class MyIntArray**

**{**

**int \* data;**

**size\_t length;**

**}**

**Реализуйте необходимые методы, пользуясь правилом трех (пяти).**