**Экзаменационный лист**

|  |  |
| --- | --- |
| «\_15\_» \_июня\_ 2020 г. | дисциплина Объектно-ориентированное программирование |
| начало \_\_\_09\_\_\_:\_\_\_00\_\_\_\_ | билет № \_\_\_\_ группа ИУ6-24Б |
| окончание \_\_\_10\_\_\_:\_\_\_30\_\_\_\_ | студент К.Ю. Каташинский |
| оценка \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ | экзаменатор Г.С. Иванова |

#### 2. Шаблоны функций. Пример.

В программировании часто приходится работать с функциями, производящими одинаковые операции, но с различными типами аргументов. Поэтому изобрели механизм **шаблонов функций** - обобщенного способа описания поведения функций, которые могут вызываться для объектов разных типов.

**Особенности шаблонов функций в С++**

* шаблон функций определяется только один раз, но это описание содержит используемые параметры. Так, для задания параметров используется конструкция вида:

**template <Список параметров шаблона> <Описание функции>;**

* каждый формальный параметр шаблона обозначается ключевым словом **class** или **typename**, за которым следует имя параметра;
* использование шаблона функции позволяет передать в нее в качестве параметра тип используемых данных, а далее выполнять операции, предусмотренные алгоритмом над объектами заданных типов
* каждый аргумент шаблона функции должен определять тип как минимум одного аргумента описываемой функции

Стоит отметить, что если для некоторых типов объектов операции, используемые в функции, не определены, следует ввести явное описание функции для этого типа.

***Пример***

Создадим аналог функции abs – модуля из стандартной библиотеки.

*#include <iostream>*

*template<typename T>*

*T abs(T x)*

*{*

*return x > 0 ? x : -x;*

*}*

*int main()*

*{*

*std::cout << "double: " << abs(-5.2) << std::endl;*

*std::cout << "int: " << abs(5) << std::endl;*

*}*