# Templates

### Ryan Baker

### January 3, 2025

## Contents

1	Introduction to Templates	2
	1.1 How Do Templates Work?	. 2
<b>2</b>	Function Templates	2
	2.1 Implicit Template Deduction	. 2
	2.2 Template Function Overloading	
	2.3 Function Template Specialization	
3	Class Templates	2
	3.1 Template Instantiation	. 2
	3.2 Class Template Specialization	. 2
4	Non-Type Template Parameters	2
5	Variadic Templates	2

### 1 Introduction to Templates

A template is a *very* powerful tool in C++. The basic idea is to use datatypes as parameters and have the compiler generate the relevant code for us. For example, you may want to write a function sort() that works for different datatypes. Rather than writing and maintaining multiple sort() functions, we can write a single sort() template and pass the datatype as a parameter.

#### 1.1 How Do Templates Work?

Templates are expanded at compile time similar to macros. The difference is that the compiler does type checking before template expansion.

#### 2 Function Templates

- 2.1 Implicit Template Deduction
- 2.2 Template Function Overloading
- 2.3 Function Template Specialization
- 3 Class Templates
- 3.1 Template Instantiation
- 3.2 Class Template Specialization
- 4 Non-Type Template Parameters
- 5 Variadic Templates