содержание

Содержание III

Список таблиц XI

Список рисунков XV

Предисловие XVI

1 Общие сведения 1

1.1 Область применения. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 1

1.2 Нормативные ссылки. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 1

1.3 Термины и определения. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 2

1.4 Соответствие реализации. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 5

1.5 Структура данного международного стандарта. , , , , , , , , , , , , , , , , , , , , , , , , , , , , 5

1.6 Синтаксис обозначения. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 6

1.7 C ++ модель памяти. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 6

1.8 C ++ объектной модели. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 7

1.9 Выполнение программы. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 8

1.10 многопоточный казней и данных гонок. , , , , , , , , , , , , , , , , , , , , , , , , , , , 11

1.11 Выражение признательности. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 15

2 Лексические конвенции 16

2.1 Отдельный перевод. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 16

2.2 Этапы перевода. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 16

2.3 Наборы символов. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 17

2.4 триграф последовательности. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 18

2.5 Препроцессирование жетоны. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 19

2.6 Альтернативные маркеры. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 20

2.7 Лексемы. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 20

2.8 Комментарии. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 20

2.9 Имена заголовков. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 20

2.10 Предварительная обработка чисел. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 21

2.11 идентификаторами. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 21

2.12 Ключевые слова. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 22

2.13 Операторы и знаки пунктуации. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 22

2.14 литералов. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 23

3 Основные понятия 32

3.1 Удостоверения и определения. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 32

3.2 Одно из правил определения. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 34

3.3 Область применения. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 37

3.4 Имя поиска. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 42

3.5 Программа и связь. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 56

3.6 Начало и окончание. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 59

3.7 Продолжительность хранения. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 62

3.8 Время жизни объекта. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 66

содержание

с ISO / IEC 2014 - Все права защищены III

ISO / IEC 14882: 2014 (E)

3.9 Типы. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 69

3.10 Lvalues ​​и rvalues. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 75

3.11 Выравнивание. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 76

4 Стандартные преобразования 78

4.1 Lvalue к Rvalue преобразования. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 79

4.2 Массив к указателю преобразования. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 79

4.3 Функция к указателю преобразования. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 79

4.4 преобразования квалификации. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 80

4.5 Интегральные акции. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 81

4.6 Плавающий продвижение точки. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 81

4.7 Интегральные преобразования. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 81

4.8 преобразования с плавающей запятой. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 82

4.9 Вещественные интегральные преобразования. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 82

4.10 преобразования указателя. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 82

4.11 Указатель преобразования членов. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 82

4.12 Булевы преобразования. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 83

4.13 Целое преобразование ранга. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 83

5 Выражения 84

5.1 Первичные выражения. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 87

5.2 Postfix выражения. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 97

5.3 Унарные выражения. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 108

5.4 Явное преобразование типа (отливать обозначения). , , , , , , , , , , , , , , , , , , , , , , , , , , , , 117

5.5 Указатель на член операторов. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 118

5.6 Мультипликативные операторы. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 118

5.7 Аддитивные операторы. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 119

5.8 Операторы сдвига. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 120

5.9 Операторы отношения. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 120

5.10 Операторы равенства. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 121

5.11 Побитовое И оператор. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 122

5,12 Побитовое исключающее ИЛИ оператора. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 122

5,13 поразрядное ИЛИ оператор. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 122

5,14 логический оператор. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 123

5,15 логический оператор ИЛИ. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 123

5,16 Условный оператор. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 123

5.17 Назначение и назначение составных операторов. , , , , , , , , , , , , , , , , , , , , , , , 125

5,18 Запятая оператор. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 126

5.19 Постоянные выражения. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 126

6 Заявления 130

6.1 помеченный оператор. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 130

6.2 Выражение утверждение. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 130

6.3 Составной оператор или блок. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 130

6.4 заявления отбора. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 131

6.5 Итерация заявления. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 132

6.6 Быстрый переход отчетности. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 135

6.7 Декларация заявление. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 136

6.8 Разрешение неоднозначности. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 137

7 Объявления 139

содержание

с ISO / IEC 2014 - Все права защищены IV

ISO / IEC 14882: 2014 (E)

7.1 Спецификаторы. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 140

7.2 Перечисление деклараций. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 157

7.3 пространствам имен. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 161

7.4 ASM декларация. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 173

7.5 Навеска характеристики. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 173

7.6 Атрибуты. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 176

8 Declarators 181

8.1 Имена типов. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 182

8.2 Разрешение неоднозначности. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 183

8.3 Значение declarators. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 184

8.4 Определения функций. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 196

8.5 Инициализаторы. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 199

9-х классов 214

9.1 Имена классов. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 216

9.2 Члены класса. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 218

9.3 Функции-члены. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 220

9.4 Статические члены. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 223

9.5 союзов. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 224

9.6 Битовые поля. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 226

9.7 декларации вложенного класса. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 227

9.8 декларации локального класса. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 228

9.9 Вложенные имена типов. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 228

10 Производные классы 230

10.1 Несколько базовых классов. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 231

10.2 Имя пользователя поиска. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 233

10.3 Виртуальные функции. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 236

10.4 Абстрактные классы. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 240

контроль доступа 11 член 242

11.1 спецификаторов доступа. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 243

11.2 Доступность базовых классов и членов базового класса. , , , , , , , , , , , , , , , , , , , , , , 244

11.3 Друзья. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 247

11.4 Защищенный доступ к члену. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 250

11.5 Доступ к виртуальным функциям. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 251

11.6 множественный доступ. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 251

11.7 Вложенные классы. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 251

12 Специальные функции-члены 253

12.1 Конструкторы. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 253

12.2 Временные объекты. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 255

12.3 конвертирование. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 258

12.4 деструкторы. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 260

12.5 Бесплатный магазин. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 263

12.6 Инициализация. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 265

12.7 Строительство и разрушение. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 270

12.8 Копирование и объекты, движущиеся класса. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 273

12.9 Наследование конструкторами. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 280

содержание

с ISO / IEC 2014 - Все права защищены V

ISO / IEC 14882: 2014 (E)

13 Чрезмерная 284

13.1 Перегружаемые деклараций. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 284

13.2 Декларация соответствия. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 286

13.3 Разрешение перегрузки. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 287

13.4 Адрес перегруженной функции. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 307

13.5 Перегруженные операторы. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 308

13.6 Встроенные операторы. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 312

14 Шаблоны 316

14.1 Параметры шаблона. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 317

14.2 Имена шаблонов специализаций. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 320

14.3 шаблона аргументы. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 322

14.4 Тип эквивалентности. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 328

14.5 Шаблон заявления. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 329

14.6 Разрешение имен. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 346

14.7 Шаблон конкретизации и специализации. , , , , , , , , , , , , , , , , , , , , , , , , , , , 359

14.8 шаблон функции специализаций. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 371

15 обработки 392 Исключение

15.1 Вызывая исключения. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 393

15.2 Конструкторы и деструкторы. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 395

15.3 Обработка исключения. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 395

15.4 спецификации исключений. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 397

15.5 Специальные функции. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 400

16 директивы Предобработка 403

16.1 Условное включение. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 404

16.2 Источник включение файла. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 405

16,3 Макро замена. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 406

16.4 линии управления. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 411

16.5 Директива об ошибке. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 412

16.6 Pragma директива. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 412

16.7 Null директива. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 412

16.8 Предопределенные имена макросов. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 412

16,9 Pragma оператора. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 413

17 Библиотека введение 414

17.1 Общие положения. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 414

17.2 Стандартная библиотека C. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 415

17.3 Определения. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 415

17.4 Дополнительные определения. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 418

17.5 Метод описания (информативный). , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 418

17.6 Библиотека в масштабах всей потребности. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 423

18 Язык: библиотека поддержки 443

18.1 Общие положения. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 443

18.2 Типы. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 443

18.3 свойства реализации. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 444

18.4 целочисленных типов. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 453

18,5 Начало и окончание. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 455

18,6 Динамическое управление памятью. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 456

содержание

с ISO / IEC 2014 - Все права защищены VI

ISO / IEC 14882: 2014 (E)

18,7 идентификации типа. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 463

18.8 Обработка исключений. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 465

18.9 инициализатор списки. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 470

18.10 Другая поддержка во время выполнения. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 471

библиотека 19 Диагностика 474

19.1 Общие положения. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 474

19.2 Исключение классов. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 474

19.3 утверждениях. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 478

19.4 Номера ошибок. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 478

Поддержка ошибки 19.5 Система. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 478

20 Общие утилиты библиотека 490

20.1 Общие положения. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 490

20.2 компоненты полезности. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 490

20.3 пар. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 495

20.4 кортежей. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 500

20.5 во время компиляции целые последовательности. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 510

20,6 Класс шаблона BitSet. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 511

20.7 памяти. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 519

20.8 Интеллектуальные указатели. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 534

20.9 Функциональные объекты. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 562

20.10 метапрограммирования и типа черты. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 584

20.11 Вкомпилировать времени рациональное арифметическое. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 603

20.12 Время коммунальные услуги. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 606

20,13 шаблон класса scoped\_allocator\_adaptor. , , , , , , , , , , , , , , , , , , , , , , , , , , 622

20.14 Класс type\_index. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 629

21 Струны библиотека 631

21.1 Общие положения. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 631

21.2 Черты характера. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 631

21.3 классы строк. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 637

21,4 Класс шаблон basic\_string. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 641

21.5 Числовые преобразования. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 669

21,6 Hash поддержка. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 671

21,7 Суффикс basic\_string литералов. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 671

21.8 нуль-терминатором утилиты последовательности. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 671

22 Локализация библиотека 675

22.1 Общие положения. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 675

22,2 Заголовок <локаль> синопсис. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 675

22.3 локалей. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 676

22.4 Стандартные локали категории. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 689

22.5 Стандартные аспекты преобразования кода. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 729

22.6 C Library локалей. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 731

23 Контейнеры библиотека 732

23.1 Общие положения. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 732

23.2 Требования к контейнеру. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 732

23.3 Последовательность контейнеров. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 760

23.4 Ассоциативные контейнеры. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 791

содержание

с ISO / IEC 2014 - Все права защищены VII

ISO / IEC 14882: 2014 (E)

23.5 Неупорядоченные ассоциативные контейнеры. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 808

23,6 контейнеров адаптеры. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 825

24 Iterators библиотека 835

24.1 Общие положения. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 835

24.2 Требования к итератора. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 835

24,3 Заголовок <итератор> синопсис. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 840

24.4 итераторов примитивов. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 843

24.5 итераторов адаптеры. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 847

24.6 поток итераторы. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 860

24,7 доступа диапазона. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 867

25 Алгоритмы библиотеки 869

25.1 Общие положения. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 869

25.2 последовательности операций Non-модифицирующие. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 880

25.3 мутирует операции последовательности. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 885

25.4 Сортировка и связанных с ними операций. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 893

25.5 алгоритмы библиотеки C. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 906

26 Числовые библиотека 908

26.1 Общие положения. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 908

26.2 Требования к числовому типу. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 908

26.3 с плавающей точкой окружающей среды. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 909

26.4 Комплексные числа. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 910

26.5 генерации случайных чисел. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 921

26.6 Числовые массивы. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 966

26.7 Обобщенные числовые операции. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 987

26,8 C библиотека. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 990

27 библиотеки ввода / вывода 995

27.1 Общие положения. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 995

27.2 Требования к Iostreams. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 995

27.3 декларации вперед. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 996

27.4 Стандартные iostream объектов. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 998

27.5 Iostreams базовые классы. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 1000

27.6 Потоковые буферы. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 1019

27.7 Форматирование и манипуляторы. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 1029

27.8 Строка на основе потоков. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 1058

27.9 на основе файлов потоков. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 1069

28 Регулярные выражения библиотеки 1085

28.1 Общие положения. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 1085

28.2 Определения. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 1085

28.3 Требования. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 1086

28,4 Заголовок <регулярное выражение> синопсис. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 1088

28,5 патезрасе :: regex\_constants. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 1095

28,6 Класс regex\_error. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 1098

28,7 шаблон класса regex\_traits. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 1098

28,8 шаблон класса basic\_regex. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 1101

28,9 Класс шаблон sub\_match. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 1108

28.10 шаблон класса match\_results. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 1114

содержание

с ISO / IEC 2014 - Все права защищены VIII

ISO / IEC 14882: 2014 (E)

28.11 алгоритмы регулярных выражений. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 1120

28.12 регулярных выражений итераторы. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 1125

28,13 Модифицированная ECMAScript регулярное выражение грамматики. , , , , , , , , , , , , , , , , , , , , , , 1131

29 атомарные операции библиотека 1134

29.1 Общие положения. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 1134

29,2 Заголовок <атомный> синопсис. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 1134

29.3 Порядок и последовательность. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 1137

29,4 безблокировочного свойство. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 1139

29.5 атомарных типов. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 1139

29.6 Операции на атомных типов. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 1143

29,7 тип флага и операции. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 1149

29.8 Заборы. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 1150

30 Тема поддержки Библиотека 1151

30.1 Общие положения. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 1151

30.2 Требования. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 1151

30.3 нити. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 1154

30.4 Взаимное исключение. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 1159

30.5 Состояние переменных. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 1179

30.6 фьючерсов. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 1187

Резюме Грамматика 1205

А.1 Ключевые слова. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 1205

A.2 Лексические конвенции. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 1205

А.3 Основные понятия. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 1209

A.4 Выражения. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , +1210

А.5 отчетности. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 1213

А.6 декларациях. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 1214

А.7 Declarators. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 1218

А.8 классов. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 1220

А.9 производных классов. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 1220

А.10 Специальные функции-члены. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 1221

П.11 Перегрузки. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 1221

A.12 шаблоны. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 1222

П.13 Обработка исключений. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 1222

A.14 директивы препроцессора. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 1223

Величины реализации B 1225

C Совместимость 1227

C.1 C ++ и ISO C. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 1227

C.2 C ++ и C ++ ISO 2003. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 1235

С.3 C ++ и C ++ ISO 2011. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 1242

С.4 C стандартная библиотека. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 1243

D Совместимость особенности 1247

D.1 оператор инкремента с BOOL операнда. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 1247

D.2 зарегистрировать ключевое слово. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 1247

D.3 неявное объявление функции копирования. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 1247

Характеристики D.4 Динамические исключение. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 1247

содержание

с ISO / IEC 2014 - Все права защищены IX

ISO / IEC 14882: 2014 (E)

D.5 C стандартная библиотека заголовков. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 1247

D.6 Старый iostreams членов. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 1248

D.7 символ \* потоки. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 1249

Д.8 Функциональные объекты. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 1258

Д.9 вяжущих. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 1262

D.10 auto\_ptr. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 1263

D.11 Нарушая исключений спецификации. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 1266

D.12 Случайные перетасовать. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 1266

E Универсальные символьные имена для символов идентификатора 1268

Е.1 Диапазоны символов допускается. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 1268

Е.2 диапазоны символов запрещен изначально. , , , , , , , , , , , , , , , , , , , , , , , , , , , , 1268

F перекрестные ссылки 1269

Индекс 1287

Индекс грамматики производств 1316

Указатель имен библиотеки 1319

Индекс определяемого реализацией поведения 1356

Список таблиц

1 триграф последовательности. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 18

2 Альтернативные жетоны. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 20

3 Идентификаторы с особым смыслом. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 22

4 Ключевые слова. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 22

5 Альтернативные представления. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 22

6 Типы целочисленных литералов. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 24

7 Escape-последовательности. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 26

8 строковый литерал конкатенации. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 29

9 Отношения константных и изменчивы. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 74

10 простой типа спецификаторы и типы они уточняют. , , , , , , , , , , , , , , , , , , , , , , , , , , 151

11 Связь между оператором и нотации вызова функции. , , , , , , , , , , , , , , , , , , , , , 292

12 конвертирование. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 300

13 Библиотека категорий. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 414

14 C ++ библиотека заголовков. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 424

15 C ++ заголовков для библиотеки объектов C. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 424

16 C ++ заголовки для автономных реализаций. , , , , , , , , , , , , , , , , , , , , , , , , , , , 425

17 EqualityComparable требования. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 426

18 LessThanComparable требования. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 426

19 DefaultConstructible требования. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 427

20 MoveConstructible требования. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 427

21 CopyConstructible требования (в дополнение к MoveConstructible). , , , , , , , , , , , , , 427

22 MoveAssignable требования. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 427

23 CopyAssignable требования (в дополнение к MoveAssignable). , , , , , , , , , , , , , , , , , 427

24 разрушаемые требования. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 427

25 Требования к NullablePointer. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 429

26 Требования к хэш-функции. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 430

27 Описательные определения переменных. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 430

28 Требования к Allocator. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 431

29 Язык Резюме Библиотека поддержки. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 443

30 Заголовок <cstddef> синопсис. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 443

31 Заголовок <climits> синопсис. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 453

32 Заголовок <cfloat> синопсис. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 453

33 Заголовок <cstdlib> синопсис. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 455

34 Заголовок <csetjmp> синопсис. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 472

35 Заголовок <csignal> синопсис. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 472

36 Заголовок <cstdalign> синопсис. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 472

37 Заголовок <cstdarg> синопсис. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 472

38 Заголовок <cstdbool> синопсис. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 473

39 Заголовок <cstdlib> синопсис. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 473

40 Заголовок <CTime> синопсис. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 473

Список таблиц

с ISO / IEC 2014 - Все права защищены XI

ISO / IEC 14882: 2014 (E)

41 Диагностика библиотека резюме. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 474

42 Заголовок <cassert> синопсис. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 478

43 Заголовок <cerrno> синопсис. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 479

44 Общие утилиты библиотека резюме. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 490

45 Заголовок <cstdlib> синопсис. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 533

46 Заголовок <CString> синопсис. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 534

47 Основные категории предикаты типа. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 588

48 Композиционные категория предикаты типа. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 589

49 Тип недвижимости предикаты. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 590

50 Вид недвижимости запросы. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 596

51 Тип отношений предикаты. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 597

52 Уст-летучее модификаций. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 598

53 Эталонные модификации. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 598

54 Знак модификации. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 599

55 массивов модификаций. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 600

56 Указатель модификаций. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 600

57 Другие преобразования. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 601

58 Выражения, используемые для выполнения арифметических операций отношения. , , , , , , , , , , , , , , , , , , , , , , , , , , , , 605

59 Требования к Тактовые. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 609

60 Заголовок <CTime> синопсис. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 622

61 Струны библиотека резюме. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 631

62 черт характера требований. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 632

63 basic\_string (Const Allocator &) эффекты. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 645

64 basic\_string (Const basic\_string &) эффекты. , , , , , , , , , , , , , , , , , , , , , , , , , , , 646

65 basic\_string (Const basic\_string &, size\_type, size\_type, сопзЬ Allocator &) эффекты. 646

66 basic\_string (Const ДИАГРАММА \*, size\_type, сопзЬ Allocator &) эффекты. , , , , , , , , , , , , , 647

67 basic\_string (Const ДИАГРАММА \*, сопзЬ Allocator &) эффекты. , , , , , , , , , , , , , , , , , , , , 647

68 basic\_string (size\_t, чарт, сопз Allocator &) эффекты. , , , , , , , , , , , , , , , , , , , 647

69 basic\_string (Const basic\_string &, сопзЬ Allocator &) и basic\_string (basic\_string &&,

Const Allocator &) эффекты. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 648

70 оператор = (Const basic\_string &) эффекты. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 648

71 оператор = (basic\_string &&) эффекты. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 649

72 (сравнить) результаты. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 663

73 Потенциальные расы mbstate\_t данных. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 673

74 Заголовок <cctype> синопсис. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 673

75 Заголовок <cwctype> синопсис. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 673

76 Заголовок <CString> синопсис. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 673

77 Заголовок <cwchar> синопсис. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 674

78 Заголовок <cstdlib> синопсис. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 674

79 Заголовок <cuchar> синопсис. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 674

80 Локализация библиотека резюме. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 675

81 Locale категория фасеты. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 679

82 Требуемые специализации. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 679

83 do\_in результат значения / do\_out. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 699

84 do\_unshift приводят значения. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 699

85 преобразования целого числа. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 703

86 Длина модификатора. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 703

87 преобразования целого числа. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 707

Список таблиц

с ISO / IEC 2014 - Все права защищены XII

ISO / IEC 14882: 2014 (E)

88 преобразования с плавающей точкой. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 707

89 Длина модификатора. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 708

90 числовых преобразований. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 708

91 Заливка обивка. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 709

92 do\_get\_date эффекты. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 716

93 Заголовок <clocale> синопсис. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 731

94 Потенциальные расы Setlocale данных. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 731

95 Контейнеры библиотека резюме. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 732

96 Требования к контейнеру. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 733

97 Реверсивные требования контейнер. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 735

98 Дополнительные контейнерные операции. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 736

99 Allocator-зависимые требования контейнер. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 738

100 Требования к последовательности контейнеров (в дополнение к контейнеру). , , , , , , , , , , , , , , , , , , , , 740

101 Факультативные контейнер последовательность операций. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 742

102 Ассоциативные требования для контейнеров (в дополнение к контейнеру). , , , , , , , , , , , , , , , , , , , 744

103 Ненумерованные ассоциативные требования контейнеров (в дополнение к таре). , , , , , , , , , , , , , 752

104 Iterators библиотека резюме. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 835

105 Отношения между итераторных категориями. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 835

106 Требования итератора. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 836

107 Входные требования итераторов (в дополнение к Iterator). , , , , , , , , , , , , , , , , , , , , , , , , 837

108 Требования Выходной итератор (в дополнение к Iterator). , , , , , , , , , , , , , , , , , , , , , , , 838

109 Вперед Требования к итераторов (в дополнение к входному итератора). , , , , , , , , , , , , , , , , , , , 839

110 двунаправленные требования итераторы (в дополнение к вперед итератора). , , , , , , , , , , , , , , , , 839

111 Требования к итераторы с произвольным доступом (в дополнение к двунаправленной итератора). , , , , , , , , , , , 840

112 Алгоритмы Резюме библиотеки. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 869

113 Заголовок <cstdlib> синопсис. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 906

114 Числовые библиотека резюме. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 908

115 Требования к последовательности Seed. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 923

116 Требования генератора случайных чисел форменные. , , , , , , , , , , , , , , , , , , , , , , , , , , 924

117 Случайные требования номер двигателя. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 925

118 Случайные требования распределения числа. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 928

119 Заголовок <CMATH> синопсис. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 991

120 Заголовок <cstdlib> синопсис. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 991

121 библиотеки ввода / вывода резюме. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 995

122 fmtflags эффекты. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 1005

123 fmtflags константы. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 1005

124 iostate эффекты. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 1005

125 OpenMode эффекты. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 1005

126 seekdir эффекты. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 1006

127 Требования к типу установки. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 1010

128 basic\_ios :: Init () эффекты. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 1012

129 basic\_ios :: copyfmt () эффекты. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , +1014

130 seekoff позиционирование. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 1062

131 newoff значения. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 1063

132 Файл открытые режимы. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 1073

133 seekoff эффекты. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 1075

Список таблиц

с ISO / IEC 2014 - Все права защищены XIII

ISO / IEC 14882: 2014 (E)

134 Заголовок <cstdio> синопсис. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 1083

135 Заголовок <cinttypes> синопсис. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 1084

136 Регулярные выражения библиотеки резюме. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 1085

137 Регулярное выражение черты требования класса. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 1086

138 syntax\_option\_type эффекты. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 1096

139 regex\_constants :: match\_flag\_type эффекты при получении матч против контейнера персонажа

Последовательность [первая, последняя). , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 1096

140 значения ERROR\_TYPE в локали C. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 1097

141 имен классов символов и соответствующие CType маски. , , , , , , , , , , , , , , , , , , , , , , , +1102

142 match\_results оператора присваивания эффектов. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 1117

143 Эффекты алгоритма алгоритма regex\_match. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 1120

144 Эффекты алгоритма regex\_search. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 1122

145 Atomics сводные библиотеки. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 1134

146 атомных интегральные. Определения типов , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 1143

147 атомных <inttypes.h>. Определения типов , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 1144

148 атомарные арифметические вычисления. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 1148

149 Тема Резюме Библиотека поддержки. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 1151

150 Стандартные макросы. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 1243

151 Стандартные значения. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 1243

152 Стандартные типы. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 1244

153 Стандартные. Структуры , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 1244

154 Стандартные функции. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 1245

155 C заголовки. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 1247

156 strstreambuf (streamsize) эффекты. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 1251

157 strstreambuf (недействительными \* (\*) (size\_t), недействительным (\*) (\*) недействительными) эффекты. , , , , , , , , , , , , , , , , , 1251

158 strstreambuf (ДИАГРАММА \*, streamsize, штурманский \*) эффекты. , , , , , , , , , , , , , , , , , , , , , , , 1251

159 seekoff позиционирования. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 1254

160 newoff значения. , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , 1254

Список таблиц

с ISO / IEC 2014 - Все права защищены XIV

# 1 Общие [вступление]

1.1 Область применения [intro.scope]

1 Настоящий стандарт устанавливает требования к реализации языка программирования C ++.

Первое такое требование состоит в том, что они реализуют язык, и поэтому настоящий стандарт также

определяет C ++. Другие требования и релаксацию первого требования появляются в разных местах в пределах

это международный стандарт.

2 C ++ является языком программирования общего назначения на основе языка программирования Си, как описано в ISO / IEC 9899: 1999 Языки программирования - С (далее в качестве стандарта C). В дополнении к

возможности, предоставляемые C, C ++ предоставляет дополнительные типы данных, классы, шаблоны, исключения, пространства имён, перегрузка оператора, имя функции перегрузкой, ссылки, операторы управления свободный магазин, а также дополнительные библиотеки объектов.

# 1.2 Нормативные ссылки [intro.refs]

1. Следующие документы являются обязательными для применения настоящего документа. Для датированных ссылок

только указанное издание. Для датированных ссылок, последнее издание указанного документа

(Включая любые поправки).

- Ecma International, спецификация ECMAScript Language, Стандартный ECMA-262, третье издание, 1999.

- ISO / IEC 2382 (все части), Информационные технологии - Словарь

- ISO / IEC 9899: 1999, Языки программирования - C

- ISO / IEC 9899: 1999 / Кор.1: 2001 (E), Языки программирования - C, Техническое исправление 1

- ISO / IEC 9899: 1999 / Cor.2: 2004 (E), Языки программирования - C, Техническое Исправление 2

- ISO / IEC 9899: 1999 / Cor.3: 2007 (E), Языки программирования - C, Техническое исправление 3

- ISO / IEC 9945: 2003, Информационные технологии - Портативный Операционная система Интерфейс (POSIX)

- ISO / IEC 10646-1: 1993, Информационные технологии -Универсал многооктетный кодированный набор символов (UCS)

- Часть 1: Архитектура и Basic Multilingual Plane

- ISO / IEC TR 19769: 2004, Информационные технологии - Языки программирования, их окружение и

системные программные интерфейсы - Расширения для языка программирования C для поддержки нового персонажа

типы данных

2 Библиотека описана в пункте 7 ISO / IEC 9899: 1999 и пунктом 7 ИСО / МЭК 9899: 1999 / Кор.1: 2001

и пунктом 7 ИСО / МЭК 9899: 1999 / Cor.2: 2003, в дальнейшем называется стандартом library.1 C

3 Библиотека описана в ISO / IEC TR 19769: 2004, в дальнейшем называется C Unicode TR.

4 Интерфейс операционной системы, описанной в ИСО / МЭК 9945: 2003 далее называется POSIX.

5 Спецификация ECMAScript Язык описано в стандарте ECMA-262, в дальнейшем называется ECMA-262.

1) С квалификации, упомянутых в пунктах 18 до 30 и в C.4, стандартная библиотека C представляет собой подмножество стандарта С ++ библиотека.

ISO / IEC 14882: 2014 (E)

# 1.3 Термины и определения [intro.defs]

1 Для целей настоящего документа применяются следующие определения.

2 17.3 определяет дополнительные термины, которые используются только в пунктах 17 по 30 и приложении D.

3 Термины, которые используются только в небольшой части этого международного стандарта определяются, где они используются и где Курсивом они определены.

1.3.1 [defns.argument]

аргумент

фактический аргумент

фактический параметр

<Выражение вызова функции> выражение в списке разделенных запятыми, ограниченной скобками

1.3.2 [defns.argument.macro]

аргумент

фактический аргумент

фактический параметр

<Функция-подобный макрос> последовательность предварительной обработки маркеров в списке разделенных запятыми, ограниченной скобками

1.3.3 [defns.argument.throw]

аргумент

фактический аргумент

фактический параметр

<Бросок выражение> операнд броска

1.3.4 [defns.argument.templ]

аргумент

фактический аргумент

фактический параметр

<Шаблон Инстанциация> выражение, тип-идентификатор или имя шаблона в списке разделенных запятыми, ограниченной

угловые скобки

1.3.5 [defns.cond.supp]

условно-поддерживается

Программа конструкт, что реализация не требуется для поддержки

[Примечание: Каждый документы по реализации все условно поддерживаемые конструкции, которые не support.-

Конец примечания]

1.3.6 [defns.diagnostic]

диагностическое сообщение

Сообщение принадлежности к реализации определенных подмножество выходных сообщений по осуществлению

1.3.7 [defns.dynamic.type]

динамический тип

<Glvalue> тип самого объекта производного (1.8), к которому glvalue обозначается glvalue выражение относится

[Пример: если указатель (8.3.1) р, чей статический тип "указатель на класс B" указывает на объект класса

D, полученный из B (пункт 10), динамический тип выражения \* р литературы "Д." (8.3.2) рассматриваются

по аналогии. -end пример]

1.3.8 [defns.dynamic.type.prvalue]

динамический тип

<Prvalue> статический тип выражения prvalue

1.3.9 [defns.ill.formed]

плохо формируется программа

программа, которая не очень хорошо сформирован

1.3.10 [defns.impl.defined]

от реализации поведение

поведение, для хорошо сформированной конструкции программы и правильные данные, что зависит от реализации и

что каждый документов реализации

1.3.11 [defns.impl.limits]

пределы осуществления

Ограничения, накладываемые на программы по реализации

1.3.12 [defns.locale.specific]

Поведение локализированных

поведение, которое зависит от местных соглашений национальности, культуры и языка, что каждая реализация

документы

1.3.13 [defns.multibyte]

многобайтный символ

последовательность из одного или нескольких байт, представляющий члена расширенного набора символов из источника или

среда выполнения

[Примечание: расширенный набор символов является расширением основного набора символов (2.3) .- конец примечание]

1.3.14 [defns.parameter]

параметр

формальный аргумент

формальный параметр

<Функция или поймать условие> объект или ссылка объявленный как часть функции объявления или определения или в

оговорка вылову обработчика исключений, который приобретает значение при входе в функцию или обработчика

1.3.15 [defns.parameter.macro]

параметр

формальный аргумент

формальный параметр

<Функция-подобный макрос> идентификатор из списка разделенных запятыми, ограниченной скобках сразу

после имени макроса

1.3.16 [defns.parameter.templ]

параметр

формальный аргумент

формальный параметр

<Шаблон> шаблон-параметр

1.3.17 [defns.signature]

подпись

§ 1.3

с ISO / IEC 2014 - Все права защищены 3

ISO / IEC 14882: 2014 (E)

<Функция> имя, список параметров типа (8.3.5), и ограждающих пространства имен (если таковые имеются)

[Примечание: Подписи используются в качестве основы для имени коверкая и linking.-конец записки]

1.3.18 [defns.signature.templ]

подпись

<Шаблон функции> имя, список параметров типа (8.3.5), вмещающих пространство имен (если таковые имеются), тип возвращаемого значения, и

Список параметров шаблона

1.3.19 [defns.signature.spec]

подпись

<Шаблон функции специализации> подпись шаблона из которого она является специализация и его шаблон

аргументы (будь то явно указано или выводится)

1.3.20 [defns.signature.member]

подпись

<Член класса функции> имя, тип параметра списка (8.3.5), класс которых функция является членом, cvqualifiers

(Если таковые имеются), и реф-классификатор (если таковые имеются)

1.3.21 [defns.signature.member.templ]

подпись

<Шаблон функции члена класса> имя, список параметров типа (8.3.5), класс которых функция является членом,

CV-классификаторы (если таковые имеются), реф-классификатор (если таковые имеются), тип возвращаемого значения и список параметров шаблона

1.3.22 [defns.signature.member.spec]

подпись

<Шаблон функции члена класса специализации> подпись шаблона функции членом которой является

специализация и ее аргументы шаблона (будь то явно указано или выводится)

1.3.23 [defns.static.type]

статический тип

Тип выражения (3.9), в результате анализа программы без учета выполнения семантики

[Примечание: Статический тип выражения зависит только от вида программы, в которой выражение

Оказывается, и не изменяется в то время как программа выполняется. -end примечание]

1.3.24 [defns.undefined]

неопределенное поведение

поведение, для которого этот международный стандарт не накладывает никаких требований

[Примечание: Неопределенное поведение можно ожидать, когда настоящий стандарт не включает каких-либо четкое определение поведение или когда программа использует ошибочную конструкцию или ошибочные данные. Допустимое неопределенное поведени варьируется от полностью игнорируя ситуацию с непредсказуемыми результатами, чтобы вести себя во время трансляции или

выполнение программы в документированной манере, характерной для окружающей среды (с или без выдачи

диагностическое сообщение), чтобы завершение перевода или выполнение (с выдачей диагностического сообщения).

Многие ошибочные программные конструкции, не порождают неопределенное поведение; они должны быть диагностированы.

-end примечание]

1.3.25 [defns.unspecified]

не определено поведение

поведение, для хорошо сформированной конструкции программы и правильные данные, что зависит от реализации

§ 1.3

с ISO / IEC 2014 - Все права защищены 4

ISO / IEC 14882: 2014 (E)

[Примечание: Реализация не требуется документ, в котором происходит поведение. Диапазон возможных

модели поведения, как правило, очерчены настоящим международным стандартом. -end примечание]

1.3.26 [defns.well.formed]

хорошо сформированные программы

Программа C ++ строится в соответствии с правилами синтаксиса, диагностируемых семантических правил, и одно из определений

Правило (3.2).

1.4 Соответствие реализации [intro.compliance]

1 Набор диагностируемых правил состоит из всех синтаксических и семантических правил в настоящем стандарте, за исключением

для тех правил, содержащих четкое обозначение, что "ни один диагностический не требуется" или которые описаны как

в результате чего "непредсказуемое поведение."

2 Хотя этот международный стандарт утверждает только требования к реализации C ++, эти требования

часто легче понять, если они сформулированы в виде требований по программам, части программ, или

выполнение программ. Такие требования имеют следующий смысл:

- Если программа не содержит никаких нарушений правил настоящего стандарта, соответствующего требованиям реализации должны, в пределах своих ресурсов, принять и правильно execute2 эту программу.

- Если программа содержит нарушение любого правила диагностируемых или возникновение конструкции, описанной в этот стандарт как "условно-поддерживается", когда реализация не поддерживает эту конструкцию,

Соответствующая реализация выдает по меньшей мере, одно диагностическое сообщение.

- Если программа содержит нарушение правила, для которых нет диагностики не требуется, это международный

Стандартные места никаких требований по реализации в отношении этой программы.

3 Для классов и шаблонов, библиотека Морозы указать частичные определения. Частные члены (пункт 11)

не указаны, но каждая реализация должна снабжать их для завершения определения в соответствии с

Описание в библиотечных пунктах.

4 Для функций, шаблонов функций, объектов и ценностей, библиотека Морозы указание деклараций. Реализации

должен поставить определения в соответствии с описаниями в библиотеке Клаусов.

5 Имена, определенные в библиотеке есть область пространства имен (7.3). единица перевода C ++ (2.2) получает доступ

к этим именам, включив соответствующий стандартный заголовок библиотеки (16.2).

6 шаблонов, классы, функции и объекты в библиотеке имеют внешнюю связь (3.5). Реализация

содержит определения стандартных библиотечных организаций, по мере необходимости, сочетая в себе единицы перевода, чтобы сформировать

полная программа C ++ (2.2).

определены

7 Два вида реализации: размещаемой реализации и отдельно стоящим реализации. Для

размещаемая реализации, настоящий стандарт определяет набор доступных библиотек. Свободностоящая

реализация в котором выполнение может происходить без использования операционной системы, и

имеет определенную реализацией набор библиотек, который включает некоторые библиотеки языка поддержки (17.6.1.3).

8 Соответствующая реализация может иметь расширения (в том числе дополнительных функций библиотеки), при условии, что они делают

не изменяет поведение любого хорошо сформированной программы. Реализации необходимы для диагностики программы,

использовать такие расширения, которые плохо сформированные в соответствии с настоящим стандартом. Сделав это, тем не менее,

они могут компилировать и выполнять такие программы.

9 Каждая реализация должна включать документацию, которая идентифицирует все условно поддерживаемые конструкции,

он не поддерживает и определяет все локализованную characteristics.3

1.5 Структура данного международного стандарта [intro.structure]

1 Статьи 2 по 16 описывают ++ язык программирования C. Это описание включает в себя подробную синтаксических

спецификации в форме, описанной в пункте 1.6. Для удобства, Приложение A повторяет все такие синтаксические характеристики.

2) "Правильное выполнение" может включать в себя неопределенное поведение, в зависимости от обрабатываемых данных; см 1.3 и 1.9.

3) Эта документация также определяет определенное реализацией поведение; см 1.9.

§ 1.5

с ISO / IEC 2014 - Все права защищены 5

ISO / IEC 14882: 2014 (E)

2 статьи 18 через 30 и Приложение D (библиотечные пункты) описывают стандартную библиотеку C ++. Это описание

включает в себя подробные описания шаблонов, классы, функции, константы и макросы, которые составляют

библиотека, в форме, описанной в пункте 17.

3 Приложение B рекомендует нижние оценки мощности соответствующих реализаций.

4 Приложение C кратко эволюцию C ++ с момента своего первого опубликованного описания, и подробно объясняет

различия между C ++ и С. О некоторых особенностях C ++ существуют исключительно для целей совместимости; Приложение D

описывает эти функции.

5 На протяжении всего этого международного стандарта, каждый пример вводится "[Пример:" и завершается

"-end Пример]". Каждая нота вводится "[Примечание:" и заканчивается "-End примечание]". Примеры и

ноты могут быть вложенными.

1.6 Синтаксис обозначения [синтаксис]

1 В синтаксической нотации, используемых в настоящем стандарте, синтаксические категории обозначаются курсивом,

и в буквальном смысле слова и символы в постоянной ширины типа. Альтернативы перечислены в отдельных строках, за исключением

несколько случаев, когда длинный набор альтернатив отмечены фразой "один из". Если текст или иной альтернативы

слишком долго, чтобы поместиться на линии, текст продолжается на последующих строках с отступом от первой. необязательный

терминал или нетерминал символ обозначается индексом "выбирают", так

{Expressionopt}

указывает на необязательный выражение, заключенное в фигурные скобки.

2 Названия для синтаксических категорий, как правило, были выбраны в соответствии со следующими правилами:

- Имя-X является использование идентификатора в контексте, который определяет его значение (например, имя класса, typedefname).

- X-идентификатор является идентификатором, не зависит от контекста значения (например, квалифицированное-ID).

- Х-сл это один или несколько крестики, не мешая разделители (например, декларация-ПОСЛ представляет собой последовательность деклараций).

- X-список представляет собой один или более крестики, разделенных запятыми вмешательства (например, выражение-список представляет собой последовательность

Выражения, разделенные запятыми).

1.7 C ++ модель памяти [intro.memory]

1 Накопитель фундаментальной в модели памяти в C ++ является байт. Байт по крайней мере, достаточно большой, чтобы содержать

любой член базового набора символов выполнения (2.3) и восемь-битовых кодовых единиц Unicode UTF-8

кодирующий форму и состоит из непрерывной последовательности битов, число которых implementationdefined.

Наименее значимый бит называется бит низкого порядка; самый старший бит называется высокого порядка

немного. Памяти, доступной для программы C ++ состоит из одной или нескольких последовательностей непрерывных байтов. каждый

байт имеет уникальный адрес.

2 [Примечание: Представление типов описана в пункте 3.9. -end примечание]

3 ячейку памяти является либо объектом скалярного типа или максимальная последовательность смежных битовых полей, имеющих все

ненулевую толщину. [Примечание: Различные особенности языка, таких как ссылки и виртуальные функции, возможно,

привлечь дополнительные ячейки памяти, которые не доступны для программ, но управляются реализацией.

-end примечание] Два или более потоков исполнения (1.10) может обновлять и доступа отдельной памяти

местах, не мешая друг другу.

4 [Примечание: Таким образом битовое поле и соседняя небитового поля находятся в отдельных ячейках памяти, и, следовательно, может быть

одновременно обновляется двух потоков исполнения без помех. То же самое относится и к двум битовых полей,

если один объявляется внутри вложенной декларации STRUCT, а другой нет, или, если два разделены

нулевой длины декларации битовое поле, или если они отделены друг от друга декларации небитового поля. Это не безопасно

одновременно обновлять два битовые поля в той же структуры, если все поля между ними также битовые поля ненулевых

ширина. -end примечание]

5 [Пример: структура объявлена

§ 1.7

с ISO / IEC 2014 - Все права защищены 6

ISO / IEC 14882: 2014 (E)

{структура

обугливается а;

INT B: 5,

C: 11,

: 0,

d: 8;

структура {INT эи: 8;} е;

}

содержит четыре отдельных ячеек памяти: Поле а и битовые поля d и e.ee каждый отдельный памяти

местоположения, и могут быть изменены одновременно, не мешая друг с другом. Битового поля Ь и с

вместе составляют четвертую ячейку памяти. Бит-поля Ь и с не могут быть одновременно изменены, но

б и, например, может быть. -end пример]

1.8 C ++ объектной модели [intro.object]

1 Конструкции в программе C ++ создавать, уничтожать, см, получать доступ и манипулировать объектами. Объект является

область хранения. [Примечание: Функция не является объектом, независимо от того, занимает ли он или нет в хранилище

так, что объекты делают. -end примечание] Объект создается с помощью определения (3.1), с помощью нового-выражения (5.3.4)

или реализацией (12.2), когда это необходимо. Свойства объекта определяются, когда объект

создано. Объект может иметь имя (пункт 3). Объект имеет длительность хранения (3.7), который влияет

его время жизни (3.8). Объект имеет тип (3.9). Тип термин объект относится к типу, с которым объект

создано. Некоторые объекты являются полиморфными (10.3); реализация генерирует информацию, связанную с

каждый такой объект, который позволяет определить тип этого объекта во время выполнения программы. Для других

объекты, интерпретация значений нашел в нем определяется типом выражений (пункт 5)

используется для доступа к ним.

2 Объекты могут содержать другие объекты, называемые подобъектов. Подобъекту может быть членом субобъект (9.2), базовая

Класс субобъект (пункт 10), или элемент массива. Объект, который не является подобъект любого другого объекта

называется полным объектом.

3 Для каждого объекта х, существует некоторый объект, называемый полный объект х, определяется следующим образом:

- Если х является законченным объектом, то х полный объект х.

- В противном случае, полный объект х полный объект (единственного) объекта, который содержит х.

4 Если полный объект, член данных (9.2), или элемент массива имеет тип класса, его тип считается

наиболее производный класс, чтобы отличить его от типа класса любого базового класса субобъекта; объектом наиболее

производного типа класса или типа внеклассовую называется наиболее производный объект.

5 Если это не битовое поле (9.6), наиболее производный объект должен иметь ненулевую размер и может занимать один или несколько

байт хранения. субобъекты базового класса может иметь нулевой размер. Объектом тривиальным Copyable или стандартного макета

типа (3.9) должны занимать смежные байт памяти.

6 Если объект не является битовое поле или базовый класс подобъектом нулевого размера, адрес этого объекта является адрес

первый байт занимает. Два объекта, которые не являются битовые поля могут иметь один и тот же адрес, если одна из них является субобъект

другого, или, если по крайней мере, один базовый класс подобъектом нулевого размера, и они бывают разных типов; в противном случае,

они должны иметь четкую addresses.4

[ Пример:

статического сопзЬ сЬаг test1 = 'х';

статического сопзЬ сЬаг test2 = 'х';

Const BOOL Ь = & = test1 & test2!; // всегда правда

-end пример]

7 [Примечание: C ++ предоставляет множество основных типов и несколько способов составления новых типов из существующих

Типы (3.9). -end примечание]

4) Под "как если бы" исключить реализацию разрешается хранить два объекта по одному адресу машины или не хранить

объект вообще, если программа не может наблюдать разницу (1.9).

§ 1.8

с ISO / IEC 2014 - Все права защищены 7

ISO / IEC 14882: 2014 (E)

исполнение 1.9 Программа [intro.execution]

1 семантические описания в настоящем стандарте определяют параметризированный недетерминированную реферат

машина. Настоящий стандарт не накладывает никаких требований о структуре соответствующих реализаций.

В частности, они не должны копировать или имитировать структуру абстрактной машины. Скорее всего, в соответствии

реализации требуются для эмуляции (только) наблюдаемое поведение абстрактной машины, как описано

below.5

2 Некоторые аспекты и операции абстрактной машины описаны в настоящем стандарте в качестве

от реализации (например, SizeOf (INT)). Они представляют собой параметры абстрактной машины.

Каждая реализация должна включать документацию с описанием его характеристик и поведения в них

respects.6 Такая документация должна определить экземпляр абстрактной машины, которая соответствует тому, что

реализация (упоминается как "по соответствующему примеру" ниже).

3 Некоторые другие аспекты и операции абстрактной машины описаны в настоящем стандарте

, как определено (например, оценка выражений в новой-инициализатора, если функция распределения не в состоянии

выделить память (5.3.4)). Там, где это возможно, настоящий стандарт определяет набор допустимых форм поведения.

Они определяют недетерминированных аспекты абстрактной машины. Экземпляр абстрактной машины может

Таким образом, есть более чем один из возможных вариантов исполнения для данной программы и данного входа.

4 Некоторые другие операции описаны в настоящем стандарте, как определено (например, эффект

в попытке изменить константный объект). [Примечание: Настоящий стандарт не устанавливает каких-либо требований по

поведение программ, которые содержат неопределенное поведение. -end примечание]

5 Соответствующая реализация выполнения хорошо сформированную программу должны производить ту же наблюдаемое поведение

в качестве одного из возможных выполнений соответствующего экземпляра абстрактной машины с той же программой

и тот же вход. Однако, если такое исполнение содержит неопределенную операцию, этот международный

Стандартные места никакое требование об осуществлении исполняющим эту программу с этим входом (даже не

с учетом операций, предшествующий первому неопределенную операцию).

6 Если обработчик сигнала выполняется в результате вызова функции поднять, то выполнение обработчика

секвенировали после вызова функции поднять и до его возвращения. [Примечание: Когда сигнал принимается

по другой причине, то выполнение обработчика сигнала, как правило, unsequenced по отношению к остальной части

Программа. -end примечание]

7 экземпляр каждого объекта с автоматической продолжительности хранения (3.7.3) связан с каждым входом в его

блок. Такой объект существует и сохраняет свою последнюю сохраненную значение во время выполнения блока и в то время как

Блок приостановлено (при вызове функции или получении сигнала).

8 Наименее требования соответствующего требованиям реализации являются:

- Доступ к изменчивых объектов оцениваются строго в соответствии с правилами абстрактной машины.

- По окончании программы, все данные, записанные в файлы должны быть идентичны одному из возможных результатов,

выполнение программы в соответствии с абстрактной семантики произвел бы.

- Динамика ввода и вывода интерактивных устройств должно осуществляться таким образом, что побудило

вывод на самом деле доставляется до того, программа ожидает ввода. Что представляет собой интерактивное устройство

от реализации.

Они все вместе обозначаются как наблюдаемое поведение программы. [Примечание: Более строгие

соответствия между абстрактным и фактической семантики может быть определена с помощью каждой реализации. -конец

заметка ]

9 [Примечание: Операторы могут быть перегруппированы в соответствии с обычными математическим правилам только тогда, когда операторы

на самом деле являются ассоциативными или commutative.7 Например, в следующем фрагменте

5) Это положение иногда называют "как если бы" правила, потому что реализация свободна игнорировать любые требования об этом

Международный стандарт до тех пор, как результат, как если бы требование было послушался, насколько это может быть определено из

наблюдаемое поведение программы. Например, фактическая реализация не должна оценивать часть выражения, если оно может

сделать вывод, что его значение не используется, и что никаких побочных эффектов, влияющих на наблюдаемое поведение программы не производятся.

6) Эта документация также включает в себя условно-поддерживаемые конструкции и поведение локализованную. См 1.4.

7) Перегруженные операторы никогда не предполагаются ассоциативными или коммутативной.

§ 1.9

с ISO / IEC 2014 - Все права защищены 8

ISO / IEC 14882: 2014 (E)

Int а, б;

/? ...? /

а = а + 32760 + Ь + 5;

выражение утверждение ведет себя точно так же, как и

а = (((а + 32760) + б) + 5);

благодаря ассоциативности и приоритете этих операторов. Таким образом, результат суммы (а + 32760) находится рядом

добавлены б, и что результат затем добавляется к 5, что приводит к значение, присвоенное. На машине, в которой

Переполнение производят исключение, и в котором диапазон значений представима в ИНТ [-32768, 32767 +],

реализация не может переписать это выражение как

а = ((а + б) + 32765);

так как если значения а и Ь были, соответственно, -32754 и -15, сумма а + Ь будет производить исключение

в то время как исходное выражение не будет; не может выражение быть переписано либо как

а = ((а + 32765) + б);

или

а = (а + (Ь + 32765));

так как значения для а и Ь, возможно, были, соответственно, 4 и -8 или -17 и 12. Однако на машине в

который переполняется не производят исключение и в которых результаты перетоков являются обратимыми, выше

Выражение утверждение может быть переписан реализации в любом из указанных выше способов, так как тот же

результат будет иметь место. -end примечание]

10 Полное выражение является выражением, которое не является подвыражением другого выражения. [Примечание: в некоторых контекстах,

такие как невычисленного операндами, синтаксический Подвыражение считается полное выражение (пункт 5). -конец

внимание] Если языковая конструкция определяется для получения неявный вызов функцию, использование языка

сооружать считается выражением для целей этого определения. Вызов деструктора

генерируется в конце срока службы объекта, отличного от временного объекта неявное полного выражения.

Конверсии применяется к результату выражения для того, чтобы удовлетворить требованиям языка

конструкт, в котором также рассматриваются появляется значок экспрессии, чтобы быть частью полного выражения.

[ Пример:

S {структура

S (INT I): I (я) {}

INT & v () {вернуться I; }

частный:

Int I;

};

S S1 (1); // Полное выражение-вызов S :: S (INT)

S S2 = 2; // Полное выражение-вызов S :: S (INT)

недействительным F () {

если (S (3) .в ()) // полное выражение включает в себя именующее к Rvalue и

// INT для преобразования BOOL, выполняется до

// Временный удаляется при завершении полного выражения

{}

}

-end пример]

11 [Примечание: Вычисление полного выражения может включать в себя оценку подвыражениям, которые не являются лексически

часть полного выражения. Например, подвыражения участвует в оценке аргументов по умолчанию (8.3.6) являются

считается созданным в выражении, который вызывает функцию, а не выражение, определяющее значение по умолчанию

аргумент. -end примечание]

§ 1.9

с ISO / IEC 2014 - Все права защищены 9

ISO / IEC 14882: 2014 (E)

12 Доступ к объекту, назначенный летучего glvalue (3.10), изменение объекта, вызывая библиотеку ввода / вывода

функции или вызова функции, которая делает любой из этих операций все побочные эффекты, которые происходят изменения в

состояние окружающей среды исполнения. Оценка выражения (или Подвыражение) в целом включает в себя

оба вычисления значений (в том числе определение идентичности объекта для оценки glvalue и загрузка

значение ранее назначенный на объект для оценки prvalue) и инициирование побочных эффектов. Когда вызов

в библиотеке ввода / вывода функция возвращает или доступ к летучим объекта оценивается побочный эффект считается

полной, даже если некоторые внешние действия подразумевается вызова (например, I / O сам по себе), либо летучего

доступ не может еще завершен.

13 Sequenced перед тем является асимметричным, транзитивным, парное соотношение между оценками выполненных одной

нить (1.10), который индуцирует частичный порядок среди этих оценок. Принимая во внимание любые две оценки А и В, если

А секвенировали перед В, то исполнение должно предшествовать выполнение B. Если A не секвенировали перед тем

Б и В не секвенирован ранее А, то А и В unsequenced. [Примечание: Выполнение unsequenced

оценки могут перекрывать друг друга. -end примечание] Оценки А и В, когда секвенировали неопределенно либо А

секвенировали перед В или В секвенированы перед тем A, но это не определено, который. [Примечание: неопределенно

секвенировали оценки не могут пересекаться, но либо может быть выполнена в первую очередь. -end примечание]

14 Каждое вычисление значения и побочный эффект, связанный с полной экспрессии секвенировали перед каждым значением

вычисление и побочный эффект, связанный со следующим полного выражения быть evaluated.8

15, за исключением особо оговоренных случаев, оценки операндам отдельных операторов и подвыражений индивидуального

выражения unsequenced. [Примечание: В выражении, который оценивается более чем один раз во время выполнения

программы, unsequenced и секвенировали оценки неопределенного ее подвыражениям не обязательно должны быть

осуществляется последовательно в разных оценках. -end примечание] Значение вычисления Операнды

Оператор упорядочиваются перед вычислением значение результата оператора. Если побочный эффект на скаляр

Объект является unsequenced по отношению к любой другой побочный эффект на той же скалярной объекта или вычисления значения

используя значение того же скалярного объекта, и они не являются потенциально параллельных (1.10), поведение

не определено. [Примечание: В следующем разделе накладывает аналогичные, но более сложные ограничения на потенциально одновременно

вычисления. -end примечание]

[ Пример:

недействительным F (INT, INT);

аннулируются г (INT I, Int \* v) {

я = v [I ++]; // Поведение не определено

я = 7, я ++, я ++; // Я становится 9

I = I ++ + 1; // Поведение не определено

I = I + 1; // Значение I увеличивается

е (я = -1, я = -1); // Поведение не определено

}

-end пример]

При вызове функции (или нет функция инлайн), каждое вычисление значения и побочный эффект

связанный с любым выражением аргумента, или с выражением постфиксном, обозначающей вызываемой функции, является

секвенировали перед выполнением каждого выражения или оператора в теле вызываемой функции. [Примечание: Значение

вычисления и побочные эффекты, связанные с различными выражениями аргумента unsequenced. -end примечание]

Каждая оценка в вызывающей функции (в том числе других вызовов функций), не иначе конкретно

секвенировали до или после выполнения тела вызываемой функции секвенировали неопределенно

уважение к исполнению называется function.9 Несколько контекстов в C ++ вызывает оценку вызова функции,

даже если не соответствующий синтаксис вызова функции не появляется в единице трансляции. [Пример: Оценка

новое выражение вызывает одну или несколько функций распределения и конструктора; 5.3.4. В качестве другого примера,

8) Как указано в 12.2, после полного вычисления выражения, последовательность из нуля или более вызовов функций деструкторов для

временные объекты происходит, как правило, в обратном порядке строительства каждого временного объекта.

9) Другими словами, функция казни не чередовать друг с другом.

§ 1.9

с ISO / IEC 2014 - Все права защищены 10

ISO / IEC 14882: 2014 (E)

вызов функции преобразования (12.3.2) может возникнуть в условиях, в которых не появляется синтаксис вызова функции.

-end пример] секвенирования ограничения на выполнение вызываемой функции (как описано выше)

1.10 многопоточный казни и данные гонки [intro.multithread]

1 поток выполнения (также известный как нить) представляет собой единый поток управления в рамках программы, включая начальный

вызов определенной функции верхнего уровня, и рекурсивно включая все функции вызова впоследствии

выполнена нитью. [Примечание: Когда один поток создает другой, первоначальный вызов функции верхнего уровня

новый поток выполняется с помощью новой нити, не создающего потока. -end примечание] Каждый поток в

Программа потенциально может получить доступ к любому объекту и функции в program.10 Под организованной реализации используется

C ++ программа может иметь более чем одну нить, которая проходит параллельно. Выполнение каждого потока доходов

как это определено в оставшейся части этого стандарта. Выполнение всей программы состоит из исполнения

все нити. [Примечание: Как правило, выполнение может рассматриваться как чередование всех его потоков. Однако,

некоторые виды атомарных операций, например, позволяют казням несогласованные с простым чередованием, как и

описано ниже. -end примечание] Под отдельно стоящей реализации, она определяется реализацией ли

программа может иметь более одного потока выполнения.

2 обработчик сигнала, который выполняется в результате вызова функции рейз принадлежит к той же нити

выполнение как вызов функции рейза. В противном случае это не определено, какой поток исполнения содержит

обработчик сигнала вызова.

3 Реализации должны обеспечить, чтобы все разблокированные потоки в конечном итоге добиться прогресса. [Примечание: Стандарт

библиотечные функции могут молча блокировать на ввода / вывода или замки. Факторы, в среде выполнения, в том числе

внешне взимаемые приоритеты потоков, может предотвратить реализацию от принятия определенных гарантий

продвижение вперед. -end примечание]

4 Казни атомных функций, которые либо определены как безблокировочного (29,7) или, как указано безблокировочного (29.4)

являются безблокировочного казни.

- Если есть только один разблокирована нить, блокировка свободной исполнение в этом потоке должен заполнить. [ Заметка:

Одновременно исполняющие потоки могут предотвратить ход выполнения выполнения безблокировочного. Например, это

ситуация может произойти с грузом автоподстройки магазине условной реализации. Это свойство иногда

непроходимость называют свободной. -end примечание]

- Когда один или более безблокировочного казни могут работать одновременно, по крайней мере, следует завершить. [ Обратите внимание

трудно для некоторых реализаций, чтобы обеспечить абсолютные гарантии на этот счет, так как повторяется и

в частности, несвоевременным помехи от других потоков может предотвратить продвижение вперед, например, путем многократного

краже строки кэша для целей, не связанных между нагрузкой и заперли в магазине условной инструкции.

Реализации должны гарантировать, что такие эффекты не могут бесконечно задерживать ход под ожидаемых рабочих

условия, и что такие аномалии могут поэтому быть безопасно игнорируются программистами. за пределами

это международный стандарт, это свойство иногда называют безблокировочного. -end примечание]

5 Значение объекта видимым для нити Т в определенный момент является начальное значение объекта, значение

присвоенное объекту через Т, или значение, присвоенное объекту другим потоком, в соответствии с правилами

ниже. [Примечание: В некоторых случаях, возможно, вместо того, чтобы быть неопределенное поведение. Большая часть этого раздела мотивировано

стремлением поддерживать атомарные операции с явными и детальных ограничений видимости. Тем не менее, он также

неявно поддерживает более простой вид для более ограниченных программ. -end примечание]

6 Два оценки экспрессии конфликт, если один из них изменяет ячейку памяти (1.7), а другой

получает доступ или изменяет ту же ячейку памяти.

7 Библиотека определяет ряд атомарных операций (пункт 29), а также операции по мьютексами (пункт 30)

которые специально определены в качестве операций синхронизации. Эти операции играют особую роль в создании

Назначения в одном потоке видимых на другой. Операция синхронизации на одну или несколько ячеек памяти

10) Объект с автоматическим или нити продолжительности хранения (3.7) связан с одним конкретным потоком, и могут быть доступны

другой поток только косвенно, через указатель или ссылку (3.9.2).

§ 1.10

с ISO / IEC 2014 - Все права защищены 11

ISO / IEC 14882: 2014 (E)

является либо потреблять операция, операция приобретают, операция высвобождения, или одновременно приобретать и выпуск

операция. Операция синхронизации без связанной ячейки памяти является забор и может быть либо

забор приобретать, забор релиз, или одновременно приобретать и выпуск забор. Кроме того, есть расслаблены атомное

операции, которые не являются операциями синхронизации, а также атомарные операции чтения-записи изменения, которые имеют

специальные характеристики. [Примечание: Например, вызов, который получает семафор будет выполнять операцию сбора по

на местах, содержащих мьютекс. Соответственно, вызов, который выпускает тот же мьютекс выполнит

релиз операции на тех же местах. Неформально, выполняя операцию версию на стороне сил предварительного

эффекты на других ячейках памяти, чтобы стать видимым для других потоков, которые позже выполнить потреблять или

приобретают операцию по А. "расслабленным" атомарные операции не являются операциями синхронизации, даже если, как и

синхронизацию операций, они не могут внести свой вклад в гонках данных. -end примечание]

8 Все модификации конкретного атомного объекта М происходят в каком-то определенном общем порядке, называется модификацию

порядок М. Если А и В являются модификацией атомного объекта М и А происходит до того (как определено ниже) B,

то А должно предшествовать B в порядке модификации М, которая определена ниже. [Примечание: Это утверждает, что

модификация заказы должны уважать "происходит до" отношений. -end примечание] [Примечание: Существует отдельная

заказ для каждого атомного объекта. Там нет требования, что они могут быть объединены в одну общую для заказа

все объекты. В целом это будет невозможно, так как различные потоки могут наблюдать изменения в разные

объекты в непоследовательных заказов. -end примечание]

9 Последовательность высвобождения во главе с операцией освобождения А на атомном объекте М является максимальным смежными подпоследовательности

побочных эффектов в порядке модификации М, где первая операция представляет собой, и каждый последующий

операция

- Выполняется одним и тем же потоком, который выполнил или

- Это атомарная операция чтения модификация-запись.

10 Некоторые библиотечные вызовы синхронизации с другими вызовами библиотеки, выполняемых другим потоком. Так, например,

атомный магазин-релиз синхронизируется с нагрузки приобретают, который берет свое значение из магазина (29.3). [ Заметка:

За исключением указанных случаев, чтение более позднее значение не обязательно обеспечивает видимость, как описано ниже.

Такое требование иногда мешают эффективной реализации. -end примечание] [Примечание:

Спецификации операций синхронизации определяют, когда кто-то читает значение, записанное другим. Для

атомных объектов, определение ясно. Все операции по данному мьютекса происходит в одном общем порядке. каждый

Приобретение мьютекс "считывает значение, записанное" последним выпуском семафоров. -end примечание]

11 Оценка А переводит зависимость к оценке B, если

- Значение А используется в качестве операнда B, если:

- В представляет собой вызов любой специализации Std :: kill\_dependency (29,3), или

- А левый операнд встроенный логическое И (&& см 5.14) или логическое ИЛИ (||, см 5.15) оператор,

или

- А левый операнд условный: оператор, или (см 5,16?)

- А левый операнд встроенной запятой (,) оператора (5.18);

или

- Это пишет скалярную объект или битового поля M, B считывает значение, написанную от М и А секвенировали

перед В, или

- Для некоторой оценки X, A несет зависимость к X, а X несет зависимость к B.

[Примечание: "Везет зависимость к" является подмножеством "секвенировали перед тем", а так же строго intrathread.

-end примечание]

12 Оценка А зависимость упорядочено перед оценки B, если

§ 1.10

с ISO / IEC 2014 - Все права защищены 12

ISO / IEC 14882: 2014 (E)

- A выполняет операцию выхода на атомном объекте М, а в другом потоке, B выполняет потреблять

операция на М и считывает значение, написанную любой побочный эффект в последовательности высвобождения во главе с А, или

- Для некоторой оценки X, А зависимость-заказано до X и X влечет за собой зависимость к B.

[Примечание: отношение "является зависимость-заказано до" аналогично "синхронизируется с", но использует релиз / -

потреблять вместо выпуска / приобрести. -end примечание]

13 Оценка Межфракционная-нить происходит перед оценки B, если

- A синхронизируется с B, или

- А зависимостей заказано до B, или

- Для некоторой оценки X

- Это синхронизируется с X и X секвенировали перед В, или

- А перед тем секвенировали X и X межпоточной происходит до B, или

- Это межпоточной происходит до X и X межпоточной происходит до B.

[Примечание: "межпоточной происходит до" отношение описывает произвольные конкатенации из "секвенировали до того»,

"Синхронизируется с" и "зависимость-заказано до" отношений, с двумя исключениями. Первый

Исключением является то, что объединение не допускается, чтобы закончить с "зависимостями заказано до", а затем "секвенировали

до". Причиной этого ограничения является то, что потребляют операцию участвует в "dependencyordered

до того "отношения предусматривает порядок только в отношении операций, к которым эта потребляющих операцию

на самом деле несет в себе зависимость. Причина, по которой это ограничение относится только к концу такого

конкатенации является то, что любая последующая операция релиз обеспечит необходимый порядок для предшествующего уровня потребления

операция. Вторым исключением является то, что конкатенация не разрешается полностью состоять из "секвенировали

до". Причины такого ограничения являются (1), чтобы разрешить "межпоточной происходит до того", чтобы быть транзитивно

закрытое и (2) "происходит до" связи, определенных ниже, предусматривает отношений, состоящих полностью из

"Секвенировали раньше". -end примечание]

14 Оценка происходит перед оценки B, если:

- А секвенировали перед В, или

- Это межпоточной происходит до B.

Реализация должна гарантировать, что ни выполнение программы не показывает цикл в "происходит до"

связь. [Примечание: Этот цикл это было бы возможно только за счет использования потреблять операций. -конец

заметка ]

15 Видимый побочный эффект А на скалярной объекта или битового поля М относительно вычисления значения Б М удовлетворяет

условия:

- Это происходит до B и

- Нет никакой другой побочный эффект Х в М такое, что А происходит до Х и Х происходит до В.

Значение неатомарном скалярного объекта или битового поля М, как определено оценки B, должно быть значение

хранится в видимой области спектра побочных эффектов А. [Примечание: Если существует неопределенность относительно которой побочный эффект неатомарном объекта

или битовое поле видна, то поведение либо не определено или не определено. -end примечание] [Примечание: Это гласит

что операции на обычных объектах не заметно заказана. Это на самом деле не обнаруживается без данных

расы, но необходимо, чтобы гарантировать, что гонки данных, как определено ниже, и с подходящими ограничениях на

использование Атомикс, соответствуют рас данных в простом чередуемой (последовательно) последовательного выполнения. -конец

заметка ]

§ 1.10

с ISO / IEC 2014 - Все права защищены 13

ISO / IEC 14882: 2014 (E)

16 Значение атомного объекта М, как определено оценки B, должно быть сохранено значение некоторой части

Эффект А, модифицирует М, где B не произойдет до А. [Примечание: Множество таких побочных эффектов также

ограничивается остальными правилами, описанными здесь, и, в частности, требованиями к согласованности ниже.

-end примечание]

17 Если операция А, модифицирует атомный объект М происходит перед операцией B, который модифицирует М, то

А должны быть раньше, чем B в порядке модификации M. [Примечание: Это требование известно как запись-запись

когерентность. -end примечание]

18 Если вычисление значение А атомного объекта М происходит до вычисления значения Б М и А принимает

его значение от побочного эффекта X на М, то значение вычисляется B должно быть либо значение, сохраненное с помощью X или

значение, сохраненное с помощью побочного эффекта Y на М, где Y следует X в порядке модификации M. [Примечание: Это

Требование известно как для чтения для чтения когерентности. -end примечание]

19 Если вычисление значение А атомного объекта М происходит перед операцией B, который модифицирует М, то

принимает свое значение из побочного эффекта X на М, где X предшествует B в порядке модификации М. [Примечание:

Это требование известно как согласованность чтения-записи. -end примечание]

20 Если побочный эффект X на атомном объекте М происходит до вычисления значения B М, то оценка

B принимает свое значение от X или от побочного эффекта Y, который следует X в порядке модификации M. [Примечание:

Это требование, как известно, от записи чтения когерентности. -end примечание]

21 [Примечание: Четыре предыдущие требования когерентности эффективно запретить компилятору изменение порядка атомных операций

к одному объекту, даже если обе операции находятся в расслабленном состоянии нагрузки. Это фактически делает когерентность кэша

гарантией обеспечиваются наиболее доступной для атомарных операций C ++ аппаратного обеспечения. -end примечание]

22 [Примечание: Значение наблюдали нагрузки атомного зависит от "происходит до" отношение, которое зависит

от значений, наблюдаемых нагрузок Атомикс. Предполагаемое чтение является то, что должно существовать объединение

атомные нагрузки с изменениями они отмечают, что, вместе с надлежащим образом выбранных заказов и модификации

К "происходит до" отношения, полученного, как описано выше, удовлетворяют ограничениям в результате, как наложены здесь.

-end примечание]

23 Два действия потенциально одновременных если

- Они выполняются разными потоками, или

- Они unsequenced, и по крайней мере один выполняется обработчиком сигнала.

Выполнение программы содержит гонки данных, если она содержит два потенциально противоречивых одновременных действий,

по крайней мере, один из которых не является атомарным, и ни происходит перед другим, для частного случая, для сигнала, за исключением

обработчики описаны ниже. Любые такие результаты гонки данные в непредсказуемое поведение. [Примечание: Можно показать, что

программы, которые правильно используют семафоры и memory\_order\_seq\_cst операции, чтобы предотвратить все гонки данных и

использовать никакие другие операции синхронизации не ведут себя, как если операции выполняются их учредительными нитями

были просто перемежаются с каждое значение вычисление объекта берутся из последнего побочного эффекта на том, что

объект в этом чередовании. Это, как правило, называют "последовательной согласованности". Тем не менее, это только относится

для данных гонки свободных программ и данных гонки свободных программ не могут наблюдать большинство преобразований программ, которые

не изменяют однопоточных семантику программы. На самом деле, большинство однопоточных преобразования программ

по-прежнему разрешено, так как любая программа, которая ведет себя по-разному в результате должен выполнить неопределенная

операция. -end примечание]

24 Два доступы к тому же объекту типа летучего sig\_atomic\_t не приводят к гонке данных, если оба происходят

в том же самом потоке, даже если один или более происходит в обработчике сигнала. Для каждого обработчика сигнала вызова,

Оценки, выполненные в потоке, ссылающегося обработчик сигнала можно разделить на две группы А и В,

таким образом, что нет оценки в B не произойдет до оценок в A, а также оценки таких летучих sig\_-

объекты atomic\_t принимают значения, как если бы все оценки в А произошло до исполнения сигнала

обработчик и выполнение обработчика сигнала происходило до всех оценок в B.

25 [Примечание: Compiler преобразования, вводящие задания к потенциально общей ячейке памяти, что

не будет изменен с помощью абстрактной машины, как правило, исключается этим стандартом, так как такое

Назначение может переписать другое задание другим потоком в тех случаях, в которых абстрактной машины

§ 1.10

с ISO / IEC 2014 - Все права защищены 14

ISO / IEC 14882: 2014 (E)

исполнение не сталкивались гонки данных. Это включает в себя реализацию присвоения элемента данных

что перезапись соседние элементы в отдельных ячейках памяти. Изменение порядка атомных нагрузок в случаях

в котором атомарных о которых идет речь, могут псевдоним, также обычно исключается, так как это может нарушить когерентность

правила. -end примечание]

26 [Примечание: Преобразования, вводящие спекулятивный читал о потенциально общей ячейке памяти не может

сохранить семантику программы C ++, как определено в настоящем стандарте, поскольку они потенциально ввести

гонки данных. Тем не менее, они, как правило, действует в контексте оптимизирующий компилятор, который нацелен конкретный

машина с четко определенными семантики для гонок данных. Они были бы недействительны для гипотетической машины,

не терпимы к гонкам или обеспечивает обнаружение аппаратных средств гонки. -end примечание]

27 Реализация может предположить, что любой поток, в конечном счете сделать одно из следующих действий:

- Прекратить,

- Сделать вызов функции библиотеки ввода / вывода,

- Доступ или изменить летучий объект, или

- Выполнить операцию синхронизации или атомарной операции.

[Примечание: Это предназначено, чтобы позволить преобразования компилятора, такие как удаление пустых петель, даже если

прекращение не может быть доказано. -end примечание]

28 Реализация должна гарантировать, что последнее значение (в порядке модификации), присвоенный атомным или

операция синхронизации станет видимым для всех других потоков в течение конечного периода времени.

1.11 Выражение признательности [intro.ack]

1 C ++ язык программирования, как описано в настоящем стандарте основывается на языке как

описано в главе R (Reference Manual) из Страуструпом: Язык программирования C ++ (второе издание,

Addison-Wesley Publishing Company, ISBN 0-201-53992-6, авторское право

с 1991 AT & T). Это, в свою очередь, базируется

на языке программирования C, как описано в Приложении А к Керниганом и Ритчи: C Программирование

Язык (Prentice-Hall, 1978, ISBN 0-13-110163-3, авторское право

с 1978 AT & T).

2 части библиотеки пунктов настоящего стандарта основаны на работе P.J. Plauger, который был

опубликован в качестве проекта стандарта C ++ Library (Prentice-Hall, ISBN 0-13-117003-1, об авторском

с 1995 P.J.

Plauger).

3 POSIX R

 является зарегистрированным товарным знаком Института инженеров по электротехнике и электронике, Inc.

4 Все права на эти оригиналы защищены.

§ 1.11

с ISO / IEC 2014 - Все права защищены 15

ISO / IEC 14882: 2014 (E)

2 Лексические конвенции [Лекса]

2,1 Отдельный перевод [lex.separate]

1 Текст программы хранится в единицах, называемых исходных файлов в настоящем стандарте. Исходный файл

вместе со всеми заголовками (17.6.1.2) и исходных файлов, включенных (16.2) с помощью директивы препроцессора

#include, за вычетом строк исходного текста пропущен какой-либо из условного включения (16.1) директивы препроцессора, является

называется единицей трансляции. [Примечание: Программа C ++ не нужно все переводить в то же время. -end примечание]

2 [Примечание: Ранее переведены единицы перевода и конкретизация единицы могут быть сохранены по отдельности или в

библиотеки. Отдельные перевода единиц программы общения (3.5) с помощью (например) вызовы функций

чьи идентификаторы имеют внешнее связывание, манипулирование объектами, идентификаторы имеют внешнее связывание, или

манипулирование файлами данных. Перевод единиц могут быть переведены отдельно, а затем позже связаны с получением

исполняемая программа (3.5). -end примечание]

2.2 Этапы перевода [lex.phases]

1 Приоритет среди синтаксических правил перевода определяется следующим phases.11

1. Физические символы исходного файла отображаются, в определенном реализацией образом, к основному источнику

набор символов (вводя символы новой строки для отслуживших линии индикаторов) в случае необходимости. Множество физических

символы исходных файлов, принимаемых от реализации. Триграф последовательности (2.4) заменяются

соответствующими односимвольные внутренние представления. Любой исходный файл символов не в основной

набор символов исходного (2.3) заменяется-символьной имя универсального, что обозначает этот символ.

(Реализация может использовать любую внутреннюю кодировку, до тех пор, как фактический длительный характер

встречается в исходном файле, и тот же расширенный характер выражается в исходном файле как

универсально-символьного имени (то есть, используя запись \ ихххх), обрабатываются за исключением того, что то же самое, где это

замена вернулась в сыром строкового литерала.)

2. Каждый экземпляр символа обратной косой черты (\) сразу за которым следует символ новой строки удаляется,

сплайсинга линии физических источника для формирования логических строк исходного текста. Только последний обратный слэш на любом физическом источнике

линия должна иметь право быть частью такого сращивания. для сращивания, вновь стали в сыром строкового литерала исключением,

если сплайс приводит к последовательности символов, которая соответствует синтаксису универсального-символьной из названия,

поведение не определено. Исходный файл, который не является пустым и не заканчивается в новой строки,

или что заканчивается символом новой строки непосредственно предшествует символ обратной косой черты, прежде чем любой такой

сплайсинга происходит, должны быть обработаны, как если дополнительный символ новой строки были приложены к файлу.

3. Исходный файл разбивается на лексемы препроцессора (2.5) и последовательностей пробельных символов

(Включая комментарии). Исходный файл не должен заканчиваться в частичном знак предварительной обработки или в частичном комментарий.

12 Каждый комментарий заменяется одним пробелом. Новые строки символов сохраняются. Будь то

каждая непустая последовательность отличных от новой строки пробельных символов сохраняется или заменяется одним

символ пробела не определен. Процесс деления персонажам исходного файла в предобработки лексем

является контекстно-зависимой. [Пример: см обработку <в пределах директивы #include предварительной обработки.

-end пример]

4. директивы Предварительная обработка выполняются, макро вызовы расширены, а унарный оператор \_Pragma

Выражения выполняются. Если последовательность символов, которая соответствует синтаксису универсального символа-имени

11) Реализации должны вести себя так, как будто происходят эти отдельные фазы, хотя на практике различные фазы могут быть сложены

вместе.

12) Частичный маркер предобработки будет возникать из исходного файла, заканчивающегося в первой части фишку нескольких символов,

требуется последовательность терминатора символов, таких как имя-заголовка, который отсутствует закрытие "или>. Частичный комментарий

будет возникать из исходного файла, заканчивающийся с незамкнутой / \* Комментарий.

§ 2.2

с ISO / IEC 2014 - Все права защищены 16

ISO / IEC 14882: 2014 (E)

получают путем конкатенации лексем (16.3.3), поведение не определено. Директива #include Предварительная обработка

вызывает указанный файл заголовка или источник для обработки от фазы 1 до фазы 4, рекурсивно.

Все директивы препроцессора затем удаляются.

5. Каждый источник набора символов член в символьный литерал или строковый литерал, а также каждый побег

Последовательность и универсально-символьного имени в символьный литерал или несырьевых строковый литерал, преобразуется в

соответствующий элемент набора символов исполнения (2.14.3, 2.14.5); нет, если нет соответствующих

член, он преобразуется в определенном реализацией члена, кроме нулевой (широкий) character.13

6. соседнему строка лексемы, сцепляются.

7. Не Пробельные символы, разделяющие лексемы больше не значительна. Каждый маркер предварительной обработки преобразуется

в знак. (2.7). Полученные маркеры синтаксически и семантически проанализированы и переведены

в качестве единицы перевода. [Примечание: Процесс анализа и перевода маркеры могут иногда

результат один жетон заменяется последовательностью других маркеров (14.2) .- конец примечание] [Примечание: Источник

файлы, единицы перевода и переведенные единицы перевода не обязательно должны быть сохранены в виде файлов, ни необходимости

там быть любая переписка взаимно однозначное соответствие между этими объектами и любого внешнего представления.

описание концептуальной только, и не определяет какой-либо конкретной реализации. -end примечание]

8. Переведенные единицы перевода и конкретизация единицы объединены следующим образом: [Примечание: Некоторые или все

они могут быть поставлены из библиотеки. -end примечание] Каждый перевел модуль перевода рассматривается в

составить список необходимых конкретизации. [Примечание: Это может включать в себя инстанциацию, которые были

явно просил (14.7.2). -end примечание] Определения необходимых шаблонов расположены.

Она определяется реализацией ли источник единиц перевода, содержащих эти определения

требуется быть доступным. [Примечание: реализация может кодировать достаточную информацию в

в переводе единицы перевода с тем, чтобы обеспечить источник не требуется здесь. -end примечание] Все необходимое

конкретизации выполняются для получения Инстанцирование единиц. [Примечание: Они похожи на переведены

единицы перевода, но не содержат ссылок на uninstantiated шаблонов и без каких-либо определений шаблонов.

-end примечание] Программа плохо сформированным, если какой-либо конкретизации терпит неудачу.

9. Все ссылки на внешние сущности разрешаются. Библиотечные компоненты связаны между собой, чтобы удовлетворить внешние ссылки

для лиц, не определенных в текущем переводе. Весь такой выход переводчик собирается в программу

изображение, которое содержит информацию, необходимую для выполнения в своей среде выполнения.

2.3 Наборы символов [lex.charset]

1 Базовый набор символов источника состоит из 96 символов: символ пробела, управляющие символы, представляющие

горизонтальная табуляция, вертикальная табуляция, форма подачи, а также новые линии, плюс следующие 91 графических символов: 14

АБВГДЕЖЗИЙКЛМНОПРСТУФХЦЧШЩЫЭЮЯ

АБВГДЕЖЗИЙКЛМНОПРСТУФХЦЧШЩЫЭЮЯ

0 1 2 3 4 5 6 7 8 9

\_ {} [] # () <>%:; , ? \* + - / ^ & | ? ! знак равно

2 Конструкт-символьного имени универсальный обеспечивает способ назвать другие символы.

гекс-четырехъядерным:

шестнадцатеричный шестнадцатеричный шестнадцатеричный шестнадцатеричный-значный

13) Реализация не нужно конвертировать все несоответствующие символы источника к тому же характеру исполнения.

14) Глифы для членов базового набора символов источника предназначены для идентификации символов из подмножества

ISO / IEC 10646, который соответствует набору символов ASCII. Однако, так как отображение из символов исходного файла с

источник набор символов (описанный в фазе перевода 1) определяется как определяется реализацией, реализация требуется

документировать, как основные символы источника представлены в исходных файлах.

§ 2.3

с ISO / IEC 2014 - Все права защищены 17

ISO / IEC 14882: 2014 (E)

универсально-символьного имя:

\ И гекс-квад

\ U шестигранной четырехъядерный шестигранной четырехъядерный

Символ, назначенный-символьной имя универсального \ UNNNNNNNN является то, что персонаж, чей характер

Краткое наименование в ИСО / МЭК 10646 является NNNNNNNN; символ, назначенный-символьной имя универсального \ uNNNN

является то, что персонаж, характер которого сокращенное наименование в ИСО / МЭК 10646 является 0000NNNN. Если шестнадцатеричное значение для

универсально-символьного имя соответствует суррогатной точки кода (в диапазоне 0xD800-0xDFFF, включительно),

программа плохо сформирована. Кроме того, если шестнадцатеричное значение для универсального-символьной имя вне

с-символ последовательности, s-символ последовательности, или г-углеродно последовательность символа или строки литералов соответствует элементу управления

характер (ни в одном из диапазонов 0x00-0x1F или 0x7F-0x9F, оба включительно) или символа в основной

Источник набора символов, программа плохо formed.15

3 Основное исполнение набора символов и каждый набор широких символов основное исполнение должно содержать все

члены базового набора символов источника, а также управляющие символы, представляющие предупреждения, Backspace и перевозки

возврат, плюс нулевой символ (соответственно, нулевой символ), представление которой имеет все нулевые биты. Для

каждый базовый набор символов исполнения, значения членов должны быть неотрицательными и отличаются друг от

другой. В обоих основных наборов символов источника и выполнения, значение каждого символа после 0 в

Приведенный выше перечень десятичных цифр должно быть на единицу больше, чем значение предыдущего. Набор символов исполнения

и набор выполнения широких символов зависят от реализации надмножествами основного характера исполнения

установить и базовый набор выполнения широких символов, соответственно. Значения членов исполнения

наборы символов и множества дополнительных членов являются локализированных.

2.4 триграф последовательности [lex.trigraph]

1 Перед тем как любой другой обработки происходит, каждое вхождение одной из следующих последовательностей трех символов

( "триграф последовательности") заменяется на один символ, указанным в таблице 1.

Таблица 1 - триграф последовательности

Триграф Замена триграф Замена Замена триграф

?? = # ?? ([?? <{

?? / \ ??)] ??>}

?? '??! | ?? -?

2 [Пример:

?? = Определить arraycheck (а, б) ?? (б ??) ??! ??! б ?? (а ??)

становится

#define arraycheck (а, б) [б] || б [а]

-end пример]

3 Ни одна другая последовательность триграф не существует. Каждый? что не начинается один из триграфы, перечисленных выше, не является

изменилось.

15) Последовательность символов, напоминающих универсальный-символьного имени в R-полукокса-последовательности (2.14.5) не образует universalcharacter-

имя.

§ 2.4

с ISO / IEC 2014 - Все права защищены 18

ISO / IEC 14882: 2014 (E)

2.5 Предварительная обработка лексем [lex.pptoken]

Предварительная обработка-маркер:

Заголовок-имя

идентификатор

С.-номер

символьного литерала

определяемых пользователем символов литерала

Строка литерала

определяемые пользователем строка-буквальным

Предварительная обработка-оп-или-punc

каждая небелый-символ пробела, который не может быть одним из вышеперечисленных

1 Каждый маркер предварительной обработки, который преобразуется в знак (2.7) имеют лексическую форму ключевого слова, А.Н.

идентификатор, буквальным, оператор или punctuator.

2 Маркер предварительной обработки минимальный лексический элемент языка в фазах перевода с 3 по 6.

категории предварительной обработки маркера являются: имена заголовков, идентификаторы препроцессора числа, символьные литералы

(В том числе определяемые пользователем литералов символов), строковых литералов (в том числе определяемые пользователем строковых литералов), предварительная обработка

операторы и знаки пунктуации, и отдельные, не пробельные символы, которые лексически не совпадают с другой

предварительная обработка маркеров категории. Если "или" символ совпадает с последней категории, поведение не определено.

Предварительная обработка маркеры могут быть разделены пробелами; это состоит из комментариев (2.8) или бело-пространстве

символы (пробел, горизонтальная табуляция, новой строки, вертикальные вкладки и форм-FEED), или обоих. Как описано в пункте 16,

при определенных обстоятельствах во время фазы трансляции 4, белое пространство (или его отсутствие) служит еще

чем предварительная обработка маркера разделения. Белое пространство может появиться в знак предварительной обработки только как часть

имя заголовка или между символами кавычек в символьный литерал или строковый литерал.

3 Если входной поток был проанализирован в предварительной обработке лексемы до данного символа:

- Если следующий символ начинается последовательность символов, которая может быть префикс и начальное двойные кавычки из

сырым строковый литерал, такие как R ", следующий маркер предварительной обработки должен быть сырым строковый литерал. Между

начальные и конечные двойные кавычки из сырой строки, любые преобразования выполняются в фазах 1

и 2 (триграфы, универсально-символьных имен, и линия сращивания) отменяются; эта реверсия применяется

перед любым д-гольца, г-символ или разграничении скобка идентифицируется. Необработанные строка символов определяется как

самая короткая последовательность символов, которая соответствует шаблону сырой строки

кодирование-prefixoptR сырой строки

- В противном случае, если следующие три символа <:: и последующий символ не является ни: ни>, то <

трактуется как символ препроцессора сам по себе, а не в качестве первого символа альтернативного маркера <:.

- В противном случае, следующая лексема предварительная обработка является самой длинной последовательности символов, которые могут представлять собой

предварительную обработку маркера, даже если это приведет к дальнейшему лексического анализа на провал.

[ Пример:

#define R "х"

Const символ \* s = R "у"; // Плохо формируется сырая строка, а не "х" "у"

-end пример]

4 [Пример: фрагмент программы 1EX обрабатывается как знак номера предварительной обработки (тот, который не является допустимым

плавающей или целочисленный литерал маркер), даже несмотря на то синтаксического анализа как пара предварительной обработки маркеров 1 и Ex может

производят допустимое выражение (например, если Ex были макрос, определенный как +1). Аналогичным образом, фрагмент программы

1E1 обрабатывается как число предварительной обработки (тот, который является действительным с плавающей буквальным маркер), является ли Еа

имя макроса. -end пример]

5 [Пример: фрагмент программы х +++++ у анализируется как X ++ ++ + у, что, если х и у имеют целые типы,

нарушает ограничения на операторов приращения, даже несмотря на то, синтаксический анализ х ++ + ++ Y может дать правильный

выражение. -end пример]

§ 2.5

с ISO / IEC 2014 - Все права защищены 19

ISO / IEC 14882: 2014 (E)

2.6 Альтернативные маркеры [lex.digraph]

1 Альтернативные представления лексем предусмотрены для некоторых операторов и punctuators.16

2 Во всех отношениях языка, каждый альтернативный маркер ведет себя так же, соответственно, в качестве основного маркера,

за его spelling.17 кроме Множество альтернативных лексем определен в таблице 2.

Таблица 2 - Альтернативные маркеры

Альтернативный Первичный Альтернативный Первичный Альтернативный Первичный

<% {И && and\_eq & =

%>} Bitor | or\_eq | =

<: [Или || xor\_eq =

:>] XOR нет!

%: # Компл? not\_eq! =

%:%: ## Bitand &

2.7 Лексемы [lex.token]

лексема:

идентификатор

ключевое слово

буквальный

оператор

punctuator

1 Есть пять видов лексем: идентификаторы, ключевые слова, литералы, 18 операторов и другие сепараторы. Пробелы,

горизонтальные и вертикальные вкладки, переводы строк, и символ 'подача листа комментарии (все вместе, "белое пространство"), как описано

ниже, игнорируются, за исключением, поскольку они служат для разделения лексем. [Примечание: Некоторые белое пространство требуется для разделения

в противном случае соседние идентификаторы, ключевые слова, числовые литералы и альтернативные маркеры, содержащие алфавитные

персонажи. -end примечание]

2.8 Комментарии [lex.comment]

1 Символы / \* начинают комментарий, который заканчивается с символами \* /. Эти комментарии не

гнездо. Символы // начинают комментарий, который заканчивается со следующим символом новой строки. Если есть

Форма подачи или вертикальной табуляции таким комментарием, только пробельные символы должны появляться между ним

и новой строки, которая завершает свой комментарий; нет диагностики требуется. [Примечание: Комментарий символы

//, / \* И \* / не имеют никакого специального значения в пределах // комментарии и обрабатываются так же, как другие символы.

Аналогичным образом, символы комментария // и / \* не имеют никакого специального значения в пределах / \* Комментарий. -end примечание]

2.9 Имена заголовков [lex.header]

заголовок-имя:

<Ч-символ последовательности>

"Q-символ последовательности"

ч-символ последовательности:

ч-символ

ч-символ последовательности ч-символ

ч-символ:

любой член исходного набора символов, за исключением новой строки и>

16) К ним относятся "орграфы" и дополнительные зарезервированные слова. Термин "орграф" (маркер, состоящий из двух символов) не является

идеально описательным, так как один из альтернативных предобработки-токенов составляет%:%: и, конечно, несколько первичных лексемы содержат два

персонажи. Тем не менее, эти альтернативные маркеры, которые не являются лексические ключевые слова просторечии известный как "орграфах".

17) Таким образом, "stringized" значения (16.3.2) из ​​[и <: будет отличаться, сохраняя исходный орфографию, но маркеры могут

в противном случае можно свободно поменять местами.

18) литералы включают строки и характер и числовые литералы.

§ 2.9

с ISO / IEC 2014 - Все права защищены 20

ISO / IEC 14882: 2014 (E)

Q-символ последовательности:

Q-символ

Q-символ последовательности Q-символ

Q-символ:

любой член символа исходного набора, за исключением новой строки и "

1 Имя заголовка предварительной обработки маркеры должны появляться только в пределах директивы #include предварительной обработки (16.2).

последовательности в обеих формах заголовков имен отображаются в определенном реализацией способом заголовков или

внешних имен исходных файлов, как указано в 16.2.

2 Появление любого из символов 'или \ или какого-либо из последовательностей символов / \* // или в

Q-символ последовательности или ч-углеродно последовательность условно поддерживается с реализацией определенных семантики, поскольку

является появление символа "в Н-полукокса-sequence.19

2.10 Предварительная обработка чисел [lex.ppnumber]

С.-номер:

разряд

, разряд

С.-значный номер

С.-номер "разряд

С.-номер "нецифровой

С.-номер идентификатора-нецифровой

С.-е число знак

С.-номер E знак

С.-номер.

1 Препроцессирование номер лексемы лексически включают все целые буквенные маркеры (2.14.2) и все плавающие буквенные маркеры

(2.14.4).

2 Ряд предварительной обработки не имеет тип или значение; она приобретает как после успешного преобразования в

целочисленный литерал маркер или плавучая буквальным маркер.

2.11 Идентификаторы [lex.name]

идентификатор:

идентификатор-нецифровой

идентификатор идентификатор-нецифровой

идентификатор цифры

идентификатор-нецифровой:

нецифровой

универсально-символьного имени

другие реализации определенных символов

нецифровой: один из

а б в г д е Ж З И Й К Л М

п о п р с т U V W X Y Z

А Б В Г Д Е Ж З И Й К Л М

Н О П Р С Т У Ф Х Ч Ш X Y Z \_

цифра: один из

0 1 2 3 4 5 6 7 8 9

1 Идентификатор сколь угодно длинную последовательность букв и цифр. Каждый персонаж-имя-универсальный в

Идентификатор должен назначить персонаж, кодирование в ISO 10646, попадает в один из диапазонов, указанных

в Е.1. Начальный элемент не должен быть универсальным-символьного имени, обозначающий символ которого кодировка

попадает в один из диапазонов, указанных в Д.2. Прописные и строчные буквы различаются. Все персонажи

significant.20

19) Таким образом, последовательность символов, которая напоминает управляющую последовательность может привести к ошибке, можно интерпретировать как символ

соответствующая управляющая последовательность, или имеют совершенно другое значение, в зависимости от реализации.

20) В системах, в которых линкеры не могут принимать расширенные символы, может быть использована кодировка символов-имя-универсального

в формировании действительных внешних идентификаторов. Например, некоторые в противном случае не используется символ или последовательность символов может быть использовано для

кодировать \ и в универсальном-символьной имя. Расширенные символы могут производить длинный внешний идентификатор, но C ++ не

§ 2.11

с ISO / IEC 2014 - Все права защищены 21

ISO / IEC 14882: 2014 (E)

2 Идентификаторы в таблице 3, имеют особое значение, когда появляется в определенном контексте. Упоминаемый

в грамматике, эти идентификаторы используются в явном виде, а не с помощью производства грамматики идентификатора.

Если не указано иное, любая неоднозначность относительно того, имеет ли данный идентификатор особое значение разрешен

интерпретировать маркер как обычный идентификатор.

Таблица 3 - Идентификаторы со специальным значением

переопределить окончательный

3 Кроме того, некоторые идентификаторы зарезервированы для использования реализаций C ++ и стандартных библиотек (17.6.4.3.2)

и не должны использоваться в других случаях; нет диагностики требуется.

2.12 Ключевые слова [lex.key]

1 Идентификаторы, показанные в таблице 4, зарезервированы для использования в качестве ключевых слов (то есть, они безоговорочно лечить

в качестве ключевых слов в фазе 7), за исключением атрибута-маркера (7.6.1) [Примечание: Экспорт ключевое слово не используется, но

зарезервировано для будущего use.-Конец примечания]:

Таблица 4 - Ключевые слова

alignas продолжить друга зарегистрировать правда

alignof decltype Гото reinterpret\_cast попробовать

ASM по умолчанию, если возвращение ЬурейеЕ

автоматическое удаление рядный короткий TypeId

BOOL сделать Int подписанный TypeName

разорвать двойной длинный SIZEOF союз

случай dynamic\_cast изменяемые статические без знака

поймать другого пространства имен static\_assert с помощью

голец перечисление новый static\_cast виртуальный

char16\_t явная noexcept структура недействительным

char32\_t экспорт nullptr переключатель летучее

Класс Экстерн оператор шаблона wchar\_t

Const ложная частная это в то время как

constexpr Поплавок защищен thread\_local

const\_cast для общественного броска

2 Кроме того, альтернативные представления, показанные в таблице 5 для некоторых операторов и знаки пунктуации (2.6)

зарезервированы и не должны использоваться в других случаях:

Таблица 5 - Альтернативные представления

и and\_eq bitand bitor не заполнить в

not\_eq или or\_eq XOR xor\_eq

2.13 Операторы и знаки пунктуации [lex.operators]

1 Лексическое представление программ C ++ включает в себя ряд маркеров предварительной обработки, которые используются в

Синтаксис препроцессора или преобразуются в маркеры для операторов и знаки пунктуации:

устанавливают ограничение перевода на значимых символов для внешних идентификаторов. В C ++, прописные и строчные буквы считаются

различны для всех идентификаторов, в том числе внешних идентификаторов.

§ 2.13

с ISO / IEC 2014 - Все права защищены 22

ISO / IEC 14882: 2014 (E)

Предварительная обработка-оп-или-punc: один из

{} [] # ## ()

<::> <%%>%:%:%:; : ...

новый удалять? ::. . \*

+ - \* /% & | ~

! = <> + = - = \* = / =% =

= & = | = << >> >> = << = == =

<=> = && || ++ -, -> \* ->

и and\_eq bitand bitor не not\_eq заполнить в

или or\_eq XOR xor\_eq

Каждая предобработки-оп-или-punc превращается в один токен в переводе фазе 7 (2.2).

2.14 Литералы [lex.literal]

2.14.1 Виды литералов [lex.literal.kinds]

1 Есть несколько видов literals.21

буквальным:

целочисленного литерала

символьного литерала

с плавающей буквальным

Строка литерала

логическое\_выражение буквальным

стрелочный буквальным

определяемые пользователем литерала

2.14.2 Целые литералы [lex.icon]

целочисленного литерала:

десятичный буквальным целое-suffixopt

восьмеричное-буквальным целое-suffixopt

шестнадцатеричное-буквальным целое-suffixopt

двоично-буквальным целое-suffixopt

десятичного литерала:

ненулевая-значный

десятичный литерал 'неавтоматического цифра

восьмеричное-буквальным:

0 восьмеричный литерал '

выбрать восьмеричное-значные

шестнадцатеричное-буквальным:

0x шестнадцатеричный-цифровой

0X шестнадцатиричное-значный

шестнадцатеричное-буквальным 'неавтоматического шестнадцатеричный-цифровой

двоично-буквальным:

0b двоично-значный

0B двоичное-значный

двоично-буквальным 'неавтоматического двоичного разряда

ненулевой цифры: один из

1 2 3 4 5 6 7 8 9

восьмеричное-значный: один из

0 1 2 3 4 5 6 7

шестнадцатеричный-цифровой: один из

0 1 2 3 4 5 6 7 8 9

а б в г д е

A B C D E F

21) Термин "буквальным" обычно обозначает, в настоящем стандарте, эти маркеры, которые называются "константы" в ISO C.

§ 2.14.2

с ISO / IEC 2014 - Все права защищены 23

ISO / IEC 14882: 2014 (E)

двоичный символ:

01

целочисленным суффикс:

без знака суффикса дальнего suffixopt

без знака суффикса долго-долго-suffixopt

долго суффикса без знака-suffixopt

долго-долго-суффикс неподписанных-suffixopt

без знака суффикса: один из

U U

длинный суффикса: один из

л L

длинный-длинный-суффикс: один из

LL LL

1 Целочисленный литерал представляет собой последовательность цифр, которая не имеет срока или показатель степени, с дополнительным отделения одного

цитаты, которые игнорируются при определении его стоимости. Целочисленный литерал может иметь префикс, который указывает

его основания и суффикс, который определяет его тип. Лексически первая цифра последовательности цифр является наиболее

значительное. Десятичное целое, буквальным (базовая десять) начинается с цифры, отличное от 0 и состоит из последовательности

десятичных цифр. Восьмеричное целое буквальным (основание восемь) начинается с цифры 0 и состоит из последовательности

восьмеричное digits.22 Шестнадцатиричный целочисленный литерал (база шестнадцать) начинается с 0x или 0X и состоит из последовательности

шестнадцатеричных цифр, которые включают в себя десятичных цифр и буквы а через F и A через F с

десятичного значения десять через пятнадцать. Двоичное число буквальным (база двух) начинается с 0b или 0B и состоит из

последовательность двоичных цифр. [Пример: Число двенадцать можно записать 12, 014, 0xc или 0b1100.

литералы 1048576, 1'048'576, 0x100000, 0x10'0000 и 0'004'000'000 все имеют одинаковое значение. -конец

пример ]

2 Тип целого литерала является первым из соответствующего списка в таблице 6, в котором его значение может быть

представлял.

Таблица 6 - Типы целочисленных литералов

Суффикс Десятичная буквальным двоичной, восьмеричной или шестнадцатеричной буквальным

доли не имеет INT INT

долго INT неподписанных INT

долго долго долго INT INT

неподписанных долгое INT

долго долго INT

неподписанных долгое долгое INT

U или U неподписанных INT неподписанных INT

неподписанных долгое INT неподписанных долгое INT

без знака долго долго INT без знака долго долго INT

л или L долго INT долго INT

долго долго INT неподписанных долгое INT

долго долго INT

неподписанных долгое долгое INT

Как и или U неподписанных долгое INT неподписанных долгое INT

и л или L без знака долго долго ИНТ без знака долго долго INT

LL или LL долго долго долго долго INT INT

неподписанных долгое долгое INT

Как и или U неподписанных долгое долгое INT неподписанных долгое долгое INT

и LL или LL

22) Цифры 8 и 9 не являются восьмеричные цифры.

§ 2.14.2

с ISO / IEC 2014 - Все права защищены 24

ISO / IEC 14882: 2014 (E)

3 Если целое число буквальным не может быть представлено любым типом в списке и расширенного целого типа (3.9.1) может

представляют свою ценность, оно может иметь что расширенный тип целого числа. Если все типы в списке для буквальное

подписан, расширенный тип целое число должно быть подписано. Если все типы в списке для буквальное без знака, то

расширенный тип целое число без знака должно быть. Если список содержит как знаком и без знака типа, удлиненная

целого типа могут быть подписаны или без знака. Программа плохо сформированным, если один из его перевода единиц содержит

целочисленный литерал, который не может быть представлен любым из разрешенных типов.

2.14.3 Символьные литералы [lex.ccon]

символьного литерала:

'С-символ последовательности'

U 'с-символ последовательности'

U 'с-символ последовательности'

L 'с-символ последовательности'

с-символ последовательности:

с-символ

с-символ последовательности с-символ

C-символ:

любой член исходного набора символов, за исключением

одинарная кавычка ', обратный слэш \ или символ новой строки

побег последовательности

универсально-символьного имени

бежать-последовательности:

простой побег последовательности

восьмеричное-побег-последовательности

шестнадцатеричное-побег-последовательности

простой побег последовательности: один из

\ "\" \ \\?

\ А \ Ь \ е \ п \ г \ т \ v

восьмеричное-побег-последовательности:

\ Восьмеричное-значный

\ Восьмеричное-значный восьмеричное-значный

\ Восьмеричное-значный восьмеричное-значный восьмеричное-значный

шестнадцатеричное-побег-последовательности:

\ Х шестнадцатеричный-цифровой

шестнадцатеричное-побег-последовательность шестнадцатеричное-значный

1 Символьный литерал представляет собой один или несколько символов, заключенные в одинарные кавычки, как в 'х', перед которым может стоять

одна из букв U, U или L, как и в и'у ', U'z', или L'х ', соответственно. Символьный литерал, который не начинается

с U, U или L представляет собой обычный литерой, также называют ограниченностью символьного литерала. Обычный

литерой который содержит один элемент с преобразовывались представима в наборе символов исполнения имеет символьного типа,

со значением, равным цифровому значению кодирования с-полукокса в наборе символов выполнения.

обычный буквенный символ, который содержит более чем один самолет С-символ является multicharacter буквальным. multicharacter

буквальным, или обычный символ литерала, содержащего одиночный с-символ не представима в характере исполнения

множество, условно-поддерживается, имеет тип Int, и имеет определенную реализацией значение.

2 Символьный литерал, который начинается с буквы U, такие как и'у ', является литерой типа char16\_t.

значение char16\_t буквального содержит один символ с-равно его ISO 10646 значение точки кода, при условии, что

точка кода представимо с помощью одного 16-разрядного блока кода. (То есть, при условии, что является одним из основных многоязычным

Плоскость код точки.) Если значение не представимо в 16 бит, программа плохо сформированным. char16\_t

литерал, содержащий несколько символов с-плохо сформированным. Символьный литерал, который начинается с буквы U, таких как

U'z ', является литерой типа char32\_t. Значение char32\_t буквальным содержащий один с-полукокс

равна его ISO 10646 значение кода точки. Char32\_t литерал, содержащий несколько символов с-плохо сформированным.

литерой, который начинается с буквы L, такие как L'х ', является широко литерой. Широкий символов

§ 2.14.3

с ISO / IEC 2014 - Все права защищены 25

ISO / IEC 14882: 2014 (E)

Буквальное имеет тип wchar\_t.23 Значение широкоапертурного литеру, содержащий единственный с-символ имеет значение, равное

к численному значению кодирования с-полукокса в наборе выполнения широких символов, если с-полукокса

не имеет своего представительства в наборе широких символов исполнения, в этом случае значение зависит от реализации.

[Примечание: Тип wchar\_t может представлять все элементы множества широких символов исполнения (см 3.9.1).

-end примечание]. Значение широкоапертурного символьный литерал, содержащий несколько с-символов зависит от реализации.

3 Некоторые неграфических символов, одиночная кавычка ', двойные кавычки ", знак вопроса?, 24 и

обратный слэш \, могут быть представлены в соответствии с таблицей 7. Двойные кавычки "и знак вопроса?, может

быть представлены как сами по себе, или управляющие последовательности \ "и \? соответственно, но апостроф '

и обратной косой черты \ должны быть представлены управляющие последовательности \ 'и \\ соответственно. Побег последовательности

в которой персонаж после обратной косой черты не указан в таблице 7 условно-поддерживается, с

определяется реализацией семантики. Последовательность выхода определяет один символ.

Таблица 7 - Escape-последовательности

новой строки NL (LF) \ п

горизонтальная табуляция HT \ т

вертикальная табуляция VT \ v

Забой BS \ б

возврат каретки CR \ г

форма подачи FF \ F

предупредит BEL \ а

обратная косая черта \ \\

вопросительный знак ? \?

апостроф '\'

двойные кавычки "\"

восьмеричное число ооо \ ооо

шестнадцатеричное число HHH \ xhhh

4 побег \ ооо состоит из бэкслеша один, два, или три восьмеричных цифры, которые принимаются для указания

значение требуемого символа. Побег \ xhhh состоит из бэкслеша х следует один

или более шестнадцатеричных цифр, которые принимаются, чтобы задать значение требуемого символа. Там нет предела

количество цифр в шестнадцатеричной последовательности. Последовательность восьмеричных или шестнадцатеричных цифр завершается

первый символ, который не является восьмеричной цифрой или шестнадцатеричной цифрой, соответственно. Значение символа

Буквальное от реализации, если она выходит за пределы диапазона, определяемого реализацией определенной для полукокса (для

литералов без префикса), char16\_t (для литералов с приставкой 'и'), char32\_t (для литералов с приставкой 'U'), или

wchar\_t (для литералов с приставкой 'L').

5 A-символьного имя универсального переводится в кодировку, в соответствующем наборе исполнения символов, из

Персонаж по имени. Если нет такой кодировки, то-символьного имени универсалия транслированы в implementationdefined

кодирование. [Примечание: В фазе перевода 1, а-символьного имени универсального вводится всякий раз, когда фактическая

расширенный символ встречается в исходном тексте. Таким образом, все расширенные символы описаны в

условия универсального характера названий. Тем не менее, фактическая реализация компилятор может использовать свой собственный родной

набор символов, до тех пор, как одни и те же результаты. -end примечание]

2.14.4 Плавающие литералы [lex.fcon]

с плавающей буквальным:

дробно-константа показатель-partopt с плавающей suffixopt

значный-последовательность показатель частей с плавающей suffixopt

23) Они предназначены для наборов символов, где персонаж не помещается в один байт.

24) Используя последовательность для знака вопроса может избежать случайного создания триграф.

§ 2.14.4

с ISO / IEC 2014 - Все права защищены 26

ISO / IEC 14882: 2014 (E)

дробно-константа:

значный-sequenceopt. значный-последовательность

значный-последовательность.

Показатель-часть:

е signopt разряд последовательности

E signopt разряд последовательности

знак: один из

+ -

значный-последовательность:

разряд

значный-последовательность 'неавтоматического цифр

плавающей суффикс: один из

е л F L

1 Плавающий буквальным состоит из целой части, десятичной точки, в дробной части, электронной или Е, необязательно подписан

-конец

ISO / IEC 14882: 2014 (E)

-end примечание]

б

-end примечание]

А "

ISO / IEC 14882: 2014 (E)

Если

Любой другой

-конец

ISO / IEC 14882: 2014 (E)

ложный

правда

## 3 Основные понятия [базовый]

1 [Примечание: Этот пункт представляет основные концепции языка C ++. Это объясняет разницу между

объект и имя, и как они соотносятся с категориями значений для выражений. Он вводит понятия

декларации и определения и представляет понятие С ++ типа, сферы применения, связи и продолжительности хранения.

Механизмы для запуска и завершения программы обсуждаются. Наконец, этот пункт представлены

основные типы языка и перечисляет способы построения типов составных от these. конца

заметка ]

2 [Примечание: Настоящий пункт не распространяется на понятия, которые затрагивают только одну часть языка. Такие понятия

обсуждаются в соответствующих пунктах. -end примечание]

3 Объект представляет собой значение, объект, ссылки, функция, переписчик, тип, член класса, шаблон, шаблон специализации,Пространство имен, параметр пакета, или это.

4 Название является использование идентификатора (2.11), оператор-функция-ID (13.5), буквальным оператор-идентификатор (13.5.8), conversionfunction-

идентификатор (12.3.2), или шаблон-идентификатор (14.2), которое обозначает объект или этикетку (6.6.4, 6.1).

5 Каждое имя, которое обозначает объект вводится посредством декларации. Каждое имя, которое обозначает метка

введены либо с помощью Гото оператора (6.6.4) или меченого-заявление (6.1).

6 Переменная вводится в объявлении ссылки, кроме нестатической элемента данных или

объект. Название переменной, если таковая имеется, обозначает ссылку или объект.

7 Некоторые имена обозначают типы или шаблоны. В общем случае, всякий раз, когда имя встречается необходимо

определить, означает ли, что имя одного из этих объектов, прежде чем продолжать анализировать программу,

содержит его. Процесс, который определяет это называется поиск имени (3.4).

8 Два имена совпадают, если

- Они представляют собой идентификаторы, состоящие из той же последовательности символов, или

- Они являются оператор-функции идентификаторы, образованные с тем же оператором, или

- Они являются конверсионные-функция-идентификаторы, образованные с таким же типом, или

- Они являются шаблонные идентификаторы, которые относятся к тому же классу или функции (14.4), или

- Они являются имена буквальных операторов (13.5.8), сформированных с тем же самым буквальным идентификатором суффиксом.

9 имя, используемое в более чем одной единице трансляции потенциально может ссылаться на тот же объект в этих переводе

единиц в зависимости от связи (3.5) имени, указанного в каждой единице трансляции.

3.1 Удостоверения и определения [basic.def]

1 Заявление (пункт 7) можно ввести одно или несколько имен в единице трансляции или переобъявить имен

введенный предыдущими декларациями. Если да, то декларация определяет интерпретацию и атрибуты из них

имена. Заявление также может иметь последствия в том числе:

- Статическое утверждение (пункт 7),

- Контроль шаблона экземпляра (14.7.2),

- Использование атрибутов (пункт 7), и

- Ничего (в случае пустого декларирования).

§ 3.1

с ISO / IEC 2014 - Все права защищены 32

ISO / IEC 14882: 2014 (E)

2 Декларация является определение, если оно не объявляет функцию без указания тела функции (8.4), то

содержит Экстерн спецификатор (7.1.1) или рычажную specification25 (7.5) и ни инициализатора, ни ТелаФункции,

он объявляет статический член данных в определении класса (9.2, 9.4), это имя класса декларации (9.1), то

непрозрачная-перечислимый-декларация (7.2), это шаблон-параметр (14.1), он является параметром-декларация (8.3.5) в

Функция декларатор, не описатель некоторой функции четкости, или это декларация ЬурейеЕ (7.1.3),

псевдоним-декларация (7.1.3), А с использованием декларирование (7.3.3), A-static\_assert декларация (пункт 7), А.Н. attributedeclaration

(Пункт 7), пустая декларация (пункт 7), или с помощью директивы (7.3.4).

[Пример: все, кроме одного из следующих определений:

Int А; // Определяет

Экстерн сопз INT с = 1; // Определяет гр

INT F (INT х) {возвращение х + а; } // Определяет п и определяет х

S структура {Int А; INT б; }; // Определяет S, S :: A, и S :: б

X структура {// определяет X

INT х; // Определяет нестатический элемента данных х

статические INT у; // Объявляет член у статических данных

X (): х (0) {} // определяет конструктор X

};

INT X :: у = 1; // Определяет X :: Y

перечисление {вверх, вниз}; // Определяет вверх и вниз

Пространство имен N {INT d; } // Определяет N и N :: d

Пространство имен N1 = N; // Определяет N1

X Anx; // Определяет Anx

в то время как это только заявления:

Экстерн Int А; // Объявляет

Экстерн сопз INT C; // Объявляет гр

INT F (INT); // Объявляет п

S структура; // Объявляет S

ЬурейеЕ INT Int; // Объявляет Int

Экстерн X anotherX; // Объявляет anotherX

используя N :: D; // Заявляет д

-end пример]

3 [Примечание: В некоторых случаях реализации C ++ неявно определяют конструктор по умолчанию (12.1), копия

Конструктор (12.8), перемещать конструктор (12.8), копировать оператор присваивания (12.8), оператор присваивания шаг

(12,8), или деструктор (12.4) функции-члены. -end примечание] [Пример: дано

#include <строка>

C {структура

станд :: строка s; // Станд :: строка является стандартной библиотекой классов (пункт 21)

};

Int основных () {

C а;

С Ь = а;

б = а;

}

реализация будет неявно определять функции, чтобы сделать определение C, эквивалентное

C {структура

станд :: строка s;

25) образующаяся внутри рамно-прилагаемая декларация-SEQ в рычажной-спецификации не влияет ли декларация является

определение.

§ 3.1

с ISO / IEC 2014 - Все права защищены 33

ISO / IEC 14882: 2014 (E)

C (): s () {}

C (сопзЬ C & х): s (x.s) {}

C (C && х): s (static\_cast <станд :: строка &&> (x.s)) {}

//: S (станд :: ход (x.s)) {}

C & оператор = (Const C & х) {S = x.s; вернуться \* это; }

C & оператор = (C && х) {s = static\_cast <станд :: строка &&> (x.s); вернуться \* это; }

// {S = станд :: ход (x.s); вернуться \* это; }

~ C () {}

};

-end пример]

4 [Примечание: Имя класса также может быть неявно объявлен развернутую типа спецификаторов (7.1.6.3). -end примечание]

5 Программа плохо сформированным, если определение любого объекта дает объекту неполный тип (3.9).

3.2 Одно из правил определения [basic.def.odr]

1 Нет ЕП не должно содержать более одного определения любой переменной, функции, типа класса, перечисления

тип или шаблон.

2 Выражение потенциально оценивается, если оно не является невычисленного операнд (пункт 5) или подвыражение

его. Множество потенциальных результатов экспрессии, е, определяется следующим образом:

- Если е-идентификатор выражения (5.1.1), набор содержит только е.

- Если е выражение доступа члена класса (5.2.5), набор содержит возможные результаты объекта

выражение.

- Если е выражение указатель на член (5.5), чей второй операнд является постоянное выражение, множество

содержит потенциальные результаты выражения объекта.

- Если е имеет вид (e1), набор содержит возможные результаты e1.

- Если е является glvalue условное выражение (5.16), множество является объединением множеств потенциальных результатах работы

Второй и третий операнды.

- Если е является выражением запятой (5,18), набор содержит возможные результаты правого операнда.

- В противном случае, множество пусто.

3 переменная х, чье имя как потенциально вычисляемое выражение ех не УСО-используется, если применяя

Lvalue к RValue преобразования (4.1) по х дает постоянное выражение (5,19), что не вызывает любого нетривиального

функции и, если х является объектом, бывший является элементом множества потенциальных результатов экспрессии, е, где

либо именующее к Rvalue преобразования (4.1) применяется к е, или е является выражение отбрасываются значение (пункт 5).

это УСО-используется, если он появляется в качестве потенциально вычисленного выражения (в том числе в результате неявного

трансформация в теле нестатической функции члена (9.3.1)). Функция виртуального элемент УСО-используется

если он не является чистым. Функция, чье имя как потенциально оценивается выражение УСО-используется, если

уникальный результат поиска или выбранный элемент из набора перегруженных функций (3.4, 13.3, 13.4), если

это чистая виртуальная функция и его имя явно не квалифицированы. [Примечание: Это охватывает вызовы по имени

функции (5.2.2), перегрузка операторов (пункт 13), определяемые пользователем преобразования (12.3.2), функция распределения для

Размещение новых (5.3.4), а также не по умолчанию инициализации (8.5). Конструктор выбран для копирования или перемещения

объект типа класса является ODR-используется даже если вызов фактически опущены путем реализации (12.8). -end примечание]

Функция распределения или открепление для класса УСО-используется новое выражение, появляющаяся в potentiallyevaluated

выражение, как указано в 5.3.4 и 12.5. Функция открепление для класса УСО-используемый об удалении

выражение появляется в потенциально вычисленного выражения, как указано в 5.3.5 и 12.5. Не-размещение

выделение или открепление функция для класса УСО-используется определение конструктора этого класса.

неразмещении функция деаллокация для класса ODR-используемый по определению деструктор этого класса,

§ 3.2

с ISO / IEC 2014 - Все права защищены 34

ISO / IEC 14882: 2014 (E)

либо выбирается в результате поиска в точке определения виртуального деструктора (12.4) .26 Присваивание

оператор-функция в классе УСО-используется неявно определенной копией присваивания или перемещения присваивания функции

для другого класса, как указано в 12.8. Конструктор по умолчанию для класса УСО-используемый по умолчанию или инициализации

инициализации значение, как указано в разделе 8.5. Конструктор для класса УСО-используется, как указано в разделе 8.5. деструктор

для класса является ODR-используется, если он потенциально вызывается (12.4).

4 Каждая программа должна содержать ровно одно определение каждого не-инлайн функции или переменной, которая является ODR используемым

в этой программе; нет диагностики не требуется. Определение может появиться в явном виде в программе, его можно найти

в стандарте или определенной пользователем библиотеки, или (когда это уместно) неявно определяется (см 12.1, 12.4 и

12,8). Встраиваемая функция должна быть определена в каждом единице перевода, в котором он ODR-используется.

5 Точно одно определение класса требуется в единице трансляции, если класс используется таким образом, что требует,

тип класса, чтобы быть полным. [Пример: следующий полный блок перевода правильно сформирован, даже если он

никогда не определяет X:

структура X; // Объявляем X как тип STRUCT

структура X \* x1; // Использование X в формировании указателя

X \* x2; // Использование X в формировании указателя

-end пример] [Примечание: Правила для деклараций и выражений, в которых описывают контексты полный класс

Типы требуются. Тип класса T должен быть полным, если:

- Объект типа Т определяется (3.1), или

- Нестатический член данных класса типа T объявляется (9.2), или

- Т используется в качестве типа объекта или массива типа элемента в новостроющемся выражении (5.3.4), или

- Именующее к Rvalue преобразование применяется к glvalue со ссылкой на объект типа T (4.1), или

- Выражение преобразуется (явно или неявно) к типу T (пункт 4, 5.2.3, 5.2.7, 5.2.9, 5.4),

или

- Это выражение, которое не является нулевым указателем постоянным, и имеет тип, кроме пустого резюме \*, преобразуется в

указатель типа Т или без ссылки на T с помощью стандартного преобразования (пункт 4), dynamic\_cast (5.2.7)

или static\_cast (5.2.9), или

- Оператор доступа член класса применяется к выражению типа Т (5.2.5), или

- Оператор TypeId (5.2.8) или оператор SizeOf (5.3.3) применяется к операнду типа Т, или

- Функция с типом возвращаемого значения или типа аргумента типа Т определяется (3.1) или называется (5.2.2), или

- Класс с базовым классом типа Т определен (пункт 10), или

- Именующее типа Т присваивается (5.17), или

- Тип T является предметом выражения alignof (5.3.6), или

- Исключение декларация имеет тип T, ссылку на Т, или указатель на T (15,3).

-end примечание]

6 Там может быть больше, чем одно определение типа класса (пункт 9), тип перечисления (7.2), встроенная функция с

внешняя связь (7.1.2), шаблон класса (пункт 14), нестатическую шаблон функции (14.5.6), статический член данных

шаблона класса (14.5.1.3), функции члена шаблона класса (14.5.1.1) или специализации шаблона для

который некоторые параметры шаблона не указаны (14.7, 14.5.5) в программе при условии, что каждое определение

появляется в другом ЕП, и при условии определения удовлетворяют следующим требованиям. Данный

такой объект с именем D определены в более чем одной единице трансляции, то

26) Реализация не требуется вызывать распределение и освобождение функции от конструкторов или деструкторов; Однако,

это допустимая методика реализации.

§ 3.2

с ISO / IEC 2014 - Все права защищены 35

ISO / IEC 14882: 2014 (E)

- Каждое определение D состоит из той же самой последовательности лексем; а также

- В каждом определении D, соответствующие имена, посмотрел вверх в соответствии с 3.4, должны относиться к определенной сущности

в определении D, или будет относиться к той же сущности, после разрешения перегрузки (13.3) и после

согласование частичной специализации шаблона (14.8.3), за исключением того, что имя может относиться к энергонезависимым

константный объект с внутренней или никакой связи, если объект имеет тот же тип буквального во всех определениях D,

а объект инициализируется константным выражением (5.19), и объект не ODR-используется, и

объект имеет одинаковое значение во всех определениях D; а также

- В каждом определении D, соответствующие объекты должны иметь один и тот же язык связь; а также

- В каждом определении D, перегруженные операторы называют, неявные вызовы функций преобразования,

Конструкторы, оператор новые функции и оператор удаления функций, должны обращаться к той же функции,

или к функции, определенной в определении D; а также

- В каждом определении D, аргумент по умолчанию, используемый в вызове функции (явной или неявной) рассматривается как

если его маркер последовательности присутствовали в определении D; то есть аргумент по умолчанию подлежит

Требования три описанные выше (и, если аргумент по умолчанию имеет подвыражения с заданным по умолчанию

аргументы, это требование применяется рекурсивно) 0,27

- Если D класс с неявно объявленной конструктор (12.1), это как если бы конструктор неявно

определяется в каждом единице перевода, где он УСО-используется и неявного определения в каждом переводе

Блок должен вызывать тот же конструктор для базового класса или члена класса из D. [Пример:

// ЕП 1:

X {структура

X (INT);

X (INT, INT);

};

X :: X (INT = 0) {}

Класс D: общественный X {};

D d2; // X (INT) вызывается D ()

// Единица перевода 2:

X {структура

X (INT);

X (INT, INT);

};

X :: X (INT = 0, Int = 0) {}

Класс D: общественный X {}; // X (INT, INT) вызывается D ();

// D () 's неявное определение

// Нарушает ODR

-end пример]

Если D представляет собой шаблон, и определяется в более чем одной единице трансляции, то предыдущие требования должны

применяются как к именам из области видимости шаблона, используемых в определении шаблона (14.6.3), а также

зависимые имена в точке инстанцирования (14.6.2). Если определения D удовлетворяют всем этим требованиям,

то программа должна вести себя, как если бы было единое определение D. Если определения D не удовлетворяют

эти требования, то поведение не определено.

27) 8.3.6 описывает, как по умолчанию имена аргументов просматриваются.

§ 3.2

с ISO / IEC 2014 - Все права защищены 36

ISO / IEC 14882: 2014 (E)

3.3 Область применения [basic.scope]

3.3.1 Декларативные регионы и области применения [basic.scope.declarative]

1 Каждое имя вводится в какой-то части текста программы называется декларативным область, которая является самой большой частью

программы, в которой это имя является действительным, то есть, в котором это имя может быть использовано в качестве безоговорочного имени

ссылаться на тот же объект. В общем случае, каждое конкретное название действует только в пределах некоторых возможно несоприкасающийся

часть текста программы называется его объем. Для определения объема декларации, иногда удобно

для обозначения потенциальной сферы действия декларации. Объем декларации такой же, как ее потенциального объема

если потенциальный объем не содержит другое заявление одного и того же имени. В этом случае потенциальный масштаб

декларации во внутреннем (содержится) декларативной области исключены из сферы действия декларации

во внешней (содержащих) декларативной области.

2 [Пример: в

INT J = 24;

Int основных () {

INT I = J, J;

J = 42;

}

идентификатор J объявляется дважды, как имя (и используется в два раза). Декларативное область первого J включает

весь пример. Потенциальный объем первого J начинается сразу после того, что ^ и распространяется на

конец программы, но ее (фактическая) сфера исключает текст между, и}. Декларативной область

второй декларации Дж (J непосредственно перед точкой с запятой) включает в себя весь текст между {и

}, Но его потенциальный масштаб исключает провозглашение I. Объем второй декларации J такая же

в качестве своего потенциального объема. -end пример]

3 Имена, объявленные в декларации вводятся в область, в которой происходит декларация, за исключением того,

что присутствие друга спецификатора (11.3), некоторые виды использования выработанного типа спецификаторов (7.1.6.3), и

с использованием Директивам (7.3.4) изменить это общее поведение.

4 Учитывая набор деклараций в одной декларативной области, каждая из которых задает один и тот же неквалифицированное имя,

- Они все относятся к той же сущности, или все они относятся к функциям и шаблонов функций; или

- Ровно одна декларация должна объявить имя класса или имя перечисления, которое не ЬурейеЕ имя

и другие заявления должны все они относятся к той же переменной или счетчику, или все они относятся к функциям

и шаблоны функций; в этом случае имя класса или перечисления имя скрыто (3.3.10). [Примечание:

имя пространства имен или имя шаблона класса должно быть уникальным в своем декларативном регионе (7.3.2, пункт 14).

-end примечание]

[Примечание: Эти ограничения применяются к декларативной области, в которую вводится имя, которое не обязательно

так же, как регион, в котором происходит декларация. В частности, разработаны типа спецификаторы (7.1.6.3)

и друг заявления (11.3) может ввести (возможно, не видно) имя в пространстве имен вмещающей; эти

ограничения применяются к данному региону. Местные ехЬегп декларации (3.5) может ввести имя в декларативная

область, где появляется объявление, а также ввести (возможно, не видно) имя в ограждающих

Пространство имен; эти ограничения применимы к обоим регионам. -end примечание]

5 [Примечание: правила поиска имен приведены в разделе 3.4. -end примечание]

3.3.2 Точка декларации [basic.scope.pdecl]

1 Точка декларации на имя сразу после его полного описателя (пункт 8) и до его

инициализатор (если таковые имеются), за исключением случаев, указанных ниже. [ Пример:

неподписанные символ х = 12;

{Неподписанные символ х = х; }

Здесь второй х инициализируется со своим собственным (неопределенном) значения. -end пример]

2 [Примечание: имя из внешней области остается видимым вплоть до момента провозглашения имени, скрывающая

это [Пример.:

§ 3.3.2

с ISO / IEC 2014 - Все права защищены 37

ISO / IEC 14882: 2014 (E)

Const INT I = 2;

{Int I [I]; }

объявляет блок-Scope массив из двух целых чисел. -end пример] -end примечание]

3 Точка декларации для шаблона класса или класса первого объявленного класса спецификатора сразу

после идентификатора или простого-шаблона-идентификатор (если таковые имеются) в своем классе головкой (раздел 9). Суть декларации

для перечисления сразу же после того, как идентификатор (если таковой имеется), либо в его перечислимом спецификатора (7.2) или его первым

непрозрачная-перечислимый-декларация (7.2), в зависимости от того что наступит раньше. Суть декларации шаблона псевдонима или псевдоним

непосредственно следует тип идентификатор, к которому псевдониму.

4 Точка объявлении с использованием-декларации, которая не называет конструктор сразу после того, как

используя декларирование (7.3.3).

5 Точка декларации для счетчику сразу после того, как его перечислителей-определения [Пример.:

Const INT х = 12;

{перечислений {х = х}; }

Здесь переписчик х инициализируется со значением постоянных х, а именно 12. -end пример]

6 После того, как точки заявления члена класса, имя члена можно посмотреть в сферу его

класс. [Примечание: это верно, даже если класс является неполным классом. Например,

X {структура

E {перечислимый г = 16};

INT B [X :: г]; // ОК

};

-end примечание]

7 Суть декларации класса впервые заявил в развернутую типа спецификатора выглядит следующим образом:

- Для декларации формы

класс-идентификатор ключа-атрибута спецификатор-seqopt;

идентификатор объявляется имя класса в объеме, который содержит объявление, в противном случае

- Для развернутую типа спецификатора вида

Идентификатор класса ключа

если выработанную типа Спецификатор используется в Децл спецификатор-SEQ или параметр декларирования-п о наличии

функция, определенная в области видимости пространства имен, идентификатор объявляется как имя класса в пространстве имен, что

содержит декларацию; в противном случае, за исключением того, как друг декларации, идентификатор объявлен в

наименьшее пространство имен или блок-область, содержащая декларацию. [Примечание: Эти правила также применяются в

шаблоны. -end примечание] [Примечание: Другие формы типа разработанной-спецификатор не объявить новое имя,

и, следовательно, должны относиться к существующему имя-типа. См 3.4.4 и 7.1.6.3. -end примечание]

8 Точка декларации на введенном-класса имя-(раздел 9) сразу же после открытия скобки

определение класса.

9 Точка декларации для функции локального предопределенной переменной (8.4) непосредственно перед функцией тела

определения функции.

10 Точка декларации для параметра шаблона сразу после его полного-шаблона параметра.

[ Пример:

определение типа во неподписанные символ T;

шаблон <класс T

= T // поиск находит имя TYPEDEF неподписанные полукокса

, T // поиск находит параметр шаблона

N = 0> структура A {};

§ 3.3.2

с ISO / IEC 2014 - Все права защищены 38

ISO / IEC 14882: 2014 (E)

-end пример]

11 [Примечание: заявления Друг относятся к функциям или классам, которые являются членами ближайшего объемлющего пространства имен,

но они не вводить новые имена в этом пространстве имен (7.3.1.2). объявления функций в блоке сферы

и объявления переменных с экстерном спецификатора в блоке сферы относятся к заявлениям, которые являются членами

объемлющей пространство имен, но они не вводят новые имена в этой области. -end примечание]

12 [Примечание: Для точки конкретизации шаблона, см-14.6.4.1. Конец примечания]

3.3.3 Блок Сфера [basic.scope.block]

1 Имя, объявленный в блоке (6.3) является локальным для этого блока; она имеет блок сферы. Его потенциальный масштаб начинается его

пункт декларации (3.3.2) и заканчивается в конце своего блока. Переменная, объявленная в блоке сферы является локальным

переменная.

2 Потенциал Область видимости имени параметра функции (в том числе один, появляющаяся в лямбда-описателя) или

функция локального предопределенную переменную в определении функции (8.4) начинается в точке его декларации. Если

Функция имеет функцию-примерочных блокировать потенциальные возможности параметра или функции локального предопределенной переменной

заканчивается в конце последнего связанного обработчика, в противном случае он заканчивается в конце внешнего блока

функция определения. Имя параметра не может быть повторно объявлен в наружном блоке функции

определение, ни в внешнем блоке любого обработчика, связанного с функцией попробуйте-блок.

3 Название объявлено в исключении декларирования является локальным для обработчика и не должен быть повторно объявлен в

наружный блок обработчика.

4 Имена объявлены в обмен на Init-заявлении для диапазона декларирования, а также в состояние, если, в то время как, для,

и переключатель заявления являются локальными для если, в то время как, для, или переключатель заявление (в том числе Контролируемое

заявление), и не может быть повторно объявлен в последующем состоянии этого заявления, ни в самом внешнем

блок (или, если заявление, любой из наружных блоков) контролируемого заявления; 6.4 см.

3.3.4 Функция прототипа сфера [basic.scope.proto]

1 В объявлении функции, или в какой-либо функции описателя за исключением описателя определения функции (8.4),

Имена параметров (при наличии) имеют прототип функции объем, который заканчивается в конце ближайшего

вмещающих функции описатель.

3.3.5 Объем функций [basic.funscope]

1 Этикетки (6.1) имеют область видимости функции и могут быть использованы в любом месте функции, в которой они объявлены. Только

метки имеют область видимости функции.

3.3.6 пространство имен сфера [basic.scope.namespace]

1 Декларативный область пространства имен-определения является пространством имен тела. Потенциальный масштаб обозначается

оригинал-пространства имен имя конкатенация декларативных областей, созданных каждым из

пространства имен определений в той же декларативной области с этой оригинальной-пространства имен имя. Субъекты объявлены

в пространстве имен тела называются членами пространства имен, а также имена, вносимые этими декларациями

в декларативной области пространства имен называются имена членов пространства имен. Пространство имен

имя элемента имеет область пространства имен. Его потенциал сфера включает в свое пространство имен с точки именем о

Декларация (3.3.2) и далее; и для каждого с помощью директивы (7.3.4), который назначает пространство имен члена,

Потенциал области видимости участника включает в себя ту часть потенциального объема, используя директиву, которая следует

точка члена в декларации. [ Пример:

Пространство имен {N

Int я;

INT г (INT а) {возвращают; }

INT J ();

недействительным д ();

}

Пространство имен {INT л = 1; }

// Потенциал объем л от своей точки декларации

// До конца единицы трансляции

§ 3.3.6

с ISO / IEC 2014 - Все права защищены 39

ISO / IEC 14882: 2014 (E)

Пространство имен {N

INT г (символ а) {// перегружает N :: г (INT)

вернуться л + а; // Л из безымянного пространства имен

}

Int я; // Ошибка: дублирующие определение

INT J (); // OK: дублирующие объявление функции

INT J () {// OK: определение N :: J ()

возвращать г (I); // Вызов N :: г (INT)

}

INT д (); // Ошибка: различный тип возвращаемого значения

}

-end пример]

2 Член пространства имен может также упоминаться после :: разрешения области видимости оператора (5.1), примененной к названию

его имен или имя пространства имен, которое назначает пространство имен члена в А с использованием директивы;

см 3.4.3.2.

3 Наружный декларативная область единицы перевода также пространство имен, называется глобальное пространство имен.

Имя, объявленное в глобальном пространстве имен имеет глобальную область пространства имен (также называется глобальной областью). потенциальный

Объем такого названия начинается в точке его декларации (3.3.2) и заканчивается в конце блока перевода

что является его декларативный область. Имя с глобальной областью пространства имен называется глобальным именем.

3.3.7 Сфера класса [basic.scope.class]

1 Следующие правила описывают сферу имен, объявленных в классах.

1) Потенциал Область видимости имени, объявленного в классе состоит не только из декларативной региона, следующего за

пункт зовут объявления, но и всех функций органов, аргументы по умолчанию, исключений спецификаций,

и приготовиться или равно-инициализаторы нестатических элементов данных в этом классе (включая такие вещи в вложенная

классов).

2) Название N используется в классе S будет относиться к той же декларации в ее контексте и при повторной оценке в

завершенная сфера S. никаких диагностических требуется за нарушение этого правила.

3) Если изменение порядка объявления членов в классе дает альтернативный действующую программу под управлением (1) и (2),

Программа плохо формируется, никаких диагностических не требуется.

4) имя, объявленная внутри функции члена скрывает объявление того же имени сфера которого простирается

или в конце прошлого класса функция члена.

5) потенциальный объем декларации, который простирается до или после конца определения класса также распространяется

в регионах, определенных его определений членов, даже если члены определенных лексически за пределами

класс (это включает в себя статические определения элемента данных, определения вложенного класса, а также функции-члена

определения, в том числе функции члена тела и любой части декларатор части таких

определения которых следует описатель идентификатор, в том числе параметра декларирование-п и любого невыполнения обязательств

аргументы (8.3.6)) [Пример:.

ЬурейеЕ INT с;

{перечислимый я = 1};

Класс X {

символ v [I]; // Ошибка: я относится к :: I

// Но когда перепроверены является X :: я

INT F () {возвращение SizeOf (с); } // OK: X :: гр

символ с;

{перечислимый я = 2};

};

§ 3.3.7

с ISO / IEC 2014 - Все права защищены 40

ISO / IEC 14882: 2014 (E)

ЬурейеЕ полукокса \* T;

Y {структура

Т а; // Ошибка: T относится к :: T

// Но когда перепроверены является Y :: T

ЬурейеЕ долго T;

Т б;

};

ЬурейеЕ INT I;

Класс D {

ЬурейеЕ I I; // Ошибка, даже если нет переназначения не участвует

};

-end пример]

2 Имя члена класса должен использоваться только в следующих случаях:

- В рамках своего класса (как описано выше) или класс, производный (раздел 10) из своего класса,

- после . Оператор применяется к выражению вида своего класса (5.2.5) или класса, производного от его

класс,

- После того, как оператор -> применяется к указателю на объект своего класса (5.2.5) или класса, производного от его

класс,

- После того, как оператор разрешения :: сфера (5.1) применительно к имени своего класса или класса, производного от его

класс.

3.3.8 Перечень сфера [basic.scope.enum]

1 Наименование Scoped счетчику (7.2) имеет объем перечисления. Его потенциальный масштаб начинается в точке его

Декларация и заканчивается в конце перечислений спецификатора.

сфера применения параметра 3.3.9 Шаблон [basic.scope.temp]

1 Декларативный область имени параметра шаблона шаблона-шаблона параметра является наименьшим

Шаблон-список параметров, в котором было введено название.

2 Декларативный область имени параметра шаблона шаблона является наименьшим шаблон-декларация

в которой было введено название. Только имена параметров шаблона принадлежат к этой декларативной области; Любые

Другой вид имя введено декларации шаблона-декларации вместо этого вводится во

такой же декларативной область, где она будет введена в результате нешаблонном декларации одного и того же

имя. [ Пример:

Пространство имен {N

шаблон <класс T> структура A {}; // # 1

шаблон <класс U> аннулируются F (U) {} // # 2

B {структура

шаблон <класс V> друг INT г (структура C \*); // # 3

};

}

Декларативной области T, U и V являются шаблонный декларации по линии № 1, № 2 и № 3, соответственно.

Но имена A, F, G и C все они принадлежат к той же декларативной области, а именно - пространство имен-тело

Н. (г по-прежнему считаются принадлежащими к этой декларативной области, несмотря на его, спрятанных во время квалифицированных и

неквалифицированное имя поиска.) -end пример]

3 Потенциал Область видимости имени параметра шаблона начинается в точке его декларации (3.3.2) и заканчивается на

конец своей декларативной области. [Примечание: Это означает, что шаблон-параметр может быть использован в объявлении

§ 3.3.9

с ISO / IEC 2014 - Все права защищены 41

ISO / IEC 14882: 2014 (E)

последующих шаблонов параметров и их аргументов по умолчанию, но не могут быть использованы в предыдущих templateparameters

или их аргументы по умолчанию. Например,

шаблон <класс Т, Т \* р, класс U = T> класс X {/? ... /};

шаблон <класс T> аннулируются е (Т \* р = новый Т);

Это также означает, что шаблон-параметр может быть использован в спецификации базовых классов. Например,

шаблон <класс T> Класс X: общественный массив <T> {/? ... /};

шаблон <класс T> Класс Y: публика T {/? ... /};

Использование параметра шаблона в качестве базового класса подразумевает, что класс, используемый в качестве аргумента шаблона должен

быть определены и не только объявляется, когда шаблон класса конкретизируется. -end примечание]

4 Декларативный область имени параметра шаблона вложен в немедленно вмещающих

декларативная область. [Примечание: В результате шаблон-параметр скрывает любое юридическое лицо, с тем же именем в

вмещающих сфера (3.3.10). [ Пример:

ЬурейеЕ Int N;

шаблон <N X, Ьурепате N, шаблон <N Y> класс T> структура A;

Здесь X представляет собой параметр шаблона, не тип типа Int и Y представляет собой параметр шаблона не типа из

такой же тип в качестве второго параметра шаблона А. -End Пример] -end примечание]

5 [Примечание: Поскольку имя параметра шаблона не может быть повторно объявлен в своей потенциальной сферы (14.6.1), A

сфера применения шаблонного параметра часто его потенциальный масштаб. Тем не менее, это все еще возможно для параметра шаблона

имя быть скрыты; см 14.6.1. -end примечание]

3.3.10 Имя скрытие [basic.scope.hiding]

1 Имя может быть скрыт явного объявления того же имени в вложенном декларативной области или производный

класс (10,2).

2 Имя класса (9.1) или имя перечисления (7.2) могут быть скрыты по имени переменной, член данных,

Функция или переписчик объявлена ​​в том же объеме. Если класс или перечисления имя и переменная, данные

член, функции или переписчик объявлены в том же объеме (в любом порядке) с тем же именем,

класс или перечисление имя скрыто везде, где переменная, член данных, функции или имя Перечислитель является

видимым.

3 В определении функции-члена, объявление имени в блоке сферы скрывает заявление члена

класса с тем же именем; см 3.3.7. Заявление члена в производном классе (пункт 10) Шкуры

заявление члена базового класса с тем же именем; см 10.2.

4 Во время поиска имени специфицировать по имени пространства имен, декларации, которые иначе были бы сделаны

видимым с помощью, используя директиву могут быть скрыты с помощью объявлений с тем же именем в пространстве имен, содержащее

что используя директивы; см (3.4.3.2).

5 Если имя находится в области видимости и не скрыт он называется видимым.

3.4 Имя поиска [basic.lookup]

1 Правила поиска имен применяются одинаково ко всем именам (в том числе TYPEDEF названий (7.1.3), пространства имен имен (7.3),

и классово-имена (9.1)) везде, где грамматика допускает такие имена в контексте обсуждается конкретная

править. Имя поиска связывает использование имени с декларацией (3.1) этого имени. Имя поиска должен

найти однозначное заявление для имени (см 10.2). поиск имени может ассоциировать более чем одно объявление

с именем, если он находит имя должно быть имя функции; заявления называются образуют множество

перегруженных функций (13.1). Разрешение перегрузки (13.3) имеет место после того, как имя успешного завершения поиска.

Правила доступа (пункт 11) рассматриваются только один раз имя поиска и функции разрешения перегрузки (если

применимо) удалось. Только после того, как поиск имен, функция разрешения перегрузки (если это применимо) и доступа

проверки удалось перечислены атрибуты, введенные декларации именем в дальнейшем используется в выражении

обработки (пункт 5).

2 Название "ищется в контексте выражения" ищется в качестве безоговорочного имени в сферу где

выражение встречается.

§ 3.4

с ISO / IEC 2014 - Все права защищены 42

ISO / IEC 14882: 2014 (E)

3 Вводимый класса имя-класса (пункт 9) также считается членом этого класса для целей

от имени сокрытия и поиска.

4 [Примечание: 3.5 обсуждаются вопросы связи. Понятия сферы, пункт декларации и имя бегах обсуждаются

в разделе 3.3. -end примечание]

3.4.1 Безусловный поиск имен [basic.lookup.unqual]

1 Во всех случаях, перечисленных в пункте 3.4.1, в области видимости искали декларации в порядке, указанном в каждом из

соответствующие категории; поиск имени заканчивается, как только декларация найдена для имени. Если объявление не является

найден, программа плохо сформированным.

2 Заявления от имен, выдвинутый с помощью директивы становятся видимыми в пространстве имен объемлющего

что используя директивы; см 7.3.4. Для неквалифицированных правил поиска имен, описанным в пункте 3.4.1, в

заявления от имен, выдвинутого использованием-директивы, считаются членами этого ограждающих

Пространство имен.

3 поиск для неквалифицированного имя, используемое в качестве постфикса-выражения вызова функции описана в разделе 3.4.2.

[Примечание: Для целей определения (во время синтаксического анализа) выражение является ли Постфиксное выражение для функции

звоните, обычные правила поиска имен применяются. Правила в разделе 3.4.2 не оказывают никакого влияния на синтаксической интерпретации

выражения. Например,

ЬурейеЕ INT F;

Пространство имен {N

структура А {

Друг недействительным F (A &);

Оператор Int ();

недействительным г (А а) {

INT I = F (а); // Е является ЬурейеЕ, а не друг

// Функция: эквивалентна INT (а)

}

};

}

Поскольку выражение не является вызовом функции, аргумент-зависимый поиск имен (3.4.2) не

применяются и друг функции F не найден. -end примечание]

4 имя, используемое в глобальной области видимости, вне какой-либо функции, класса или пользователем объявлены имен, должны быть объявлены

до его использования в глобальном масштабе.

5 Имя, используемое в пользовательском пространстве имен объявлены вне определения какой-либо функции или класса должны быть объявлены

до его использования в этом пространстве имен или перед его использованием в пространстве имен, охватывающему пространство имен.

6 имя, используемое в определении функции следующей работы функции декларатор-id28, которая является членом

Пространство имен N (где, только с целью изложения, N может представлять глобальную область видимости) должны быть объявлены

перед его использованием в блоке, в котором он или используется в одном из своих блоков вмещающих (6.3) или, объявляется

до его использования в пространстве имен N, или, если N является вложенной имен, должен быть объявлен до его использования в одном из N-х

вмещающих пространств имен. [ Пример:

Пространство имен {

Пространство имен {N

недействительным F ();

}

}

аннулированию A :: N :: F () {

I = 5;

// Следующие области видимости искали декларации I:

// 1) наружный блок сфера A :: N :: F, перед использованием I

// 2) объем пространства имен N

28) Это относится к неквалифицированным именам, которые происходят, например, в аргумент типа или значения по умолчанию в параметре декларирование-п

или используется в теле функции.

§ 3.4.1

с ISO / IEC 2014 - Все права защищены 43

ISO / IEC 14882: 2014 (E)

// 3) объем пространства имен A

// 4) глобальный масштаб, перед определением A :: N :: F

}

-end пример]

7 имя, используемое в определении класса X вне функции члена тела, аргумент по умолчанию, exceptionspecification,

приготовиться или равно-инициализатор нестатической элемента данных или вложенного класса definition29 должны быть

объявлен в одном из следующих способов:

- До его использования в классе X или быть членом базового класса X (10.2), или

- Если X является вложенным классом класса Y (9.7), до определения X в Y, или должен быть членом основания

класс Y (этот поиск применяется, в свою очередь вмещающих классов Y 's, начиная с внутренней ограждающих

класс), 30 или

- Если Х представляет собой локальный класс (9.8) или является вложенным классом локального класса, до определения класса X в блоке

ограждающих определение класса X, или

- Если X является членом пространства имен N, или является вложенным классом класса, который является членом N, или локальный класс

или вложенный класс в локальном классе функции, которая является членом N, перед определением класса

X в пространстве имен N или в одном из вмещающих пространств имен N 's.

[ Пример:

Пространство имен М {

Класс B {};

}

Пространство имен {N

Класс Y: общественный M :: B {

Класс X {

INT A [I];

};

};

}

// Следующие области видимости искали декларации I:

// 1) сфера класса N :: Y :: X, перед использованием I

// 2) сфера класса N :: Y, перед определением N :: Y :: X

// 3) объем базового класса M N :: Y :: B в

// 4) объем пространства имен N, перед определением N :: Y

// 5) глобальный масштаб, перед определением N

-end пример] [Примечание: При поиске предварительного объявления класса или функции, введенной другу по электронной почте

Декларация, прицелы за пределами сокровенное области видимости пространства имен, не учитываются; см 7.3.1.2. -

Конец примечания] [Примечание: 3.3.7 далее описывает ограничения на использование имен в определении класса. 9.7 далее

описывает ограничения на использование имен в определениях вложенного класса. 9.8 далее описывает ограничения

об использовании имен в определениях локальных классов. -end примечание]

8 Для членов класса X, имя, используемое в теле функции члена, в споре по умолчанию, в exceptionspecification,

в брекет или равно-инициализаторе нестатической элемента данных (9.2), или в определении класса

член вне определения X, следуя декларатор-id31 участника, должен быть объявлен в одном из

следующими способами:

29) Это относится к неопределенному имени после имени класса; такое имя может быть использовано в базовом пункте, или могут быть использованы в

определение класса.

30) Этот поиск применяется ли определение X вложен в определении Y или определение Х появляется в том

Пространство имен сфера ограждающих определение Y 's (9.7).

31) То есть, неквалифицированное имя, которое происходит, например, в типе в параметре декларирование-п или в exceptionspecification.

§ 3.4.1

с ISO / IEC 2014 - Все права защищены 44

ISO / IEC 14882: 2014 (E)

- Перед его использованием в блоке, в котором он используется или в блоке ограждающей (6.3), или

- Должен быть членом класса X или быть членом базового класса X (10.2), или

- Если X является вложенным классом класса Y (9.7), должен быть членом Y, или должен быть членом базового класса Y

(Этот поиск применяется в свою очередь, к вмещающими классам Y, начиная с самого внутреннего объемлющего класса), 32 или

- Если Х представляет собой локальный класс (9.8) или является вложенным классом локального класса, до определения класса X в блоке

ограждающих определение класса X, или

- Если X является членом пространства имен N, или является вложенным классом класса, который является членом N, или локальный класс

или вложенный класс в локальном классе функции, которая является членом N, перед использованием названия,

в пространстве имен N или в одном из вмещающих пространств имен N 's.

[ Пример:

Класс B {};

Пространство имен М {

Пространство имен {N

Класс X: общественный B {

недействительным F ();

};

}

}

аннулируются M :: N :: X :: F () {

I = 16;

}

// Следующие области видимости искали декларации I:

// 1) наружный блок сфера M :: N :: X :: F, перед использованием I

// 2) сфера класса M :: N :: X

// 3) сфера M :: N :: Х базовый класс B

// 4) объем пространства имен M :: N

// 5) объем пространства имен M

// 6) глобальный масштаб, до определения M :: N :: X :: F

-end пример] [Примечание: 9.3 и 9.4 далее описывают ограничения на использование имен в функции-члене

определения. 9.7 далее описывает ограничения на использование имен в рамках вложенных классов. 9.8

далее описывает ограничения на использование имен в определениях локальных классов. -end примечание]

9 Имя поиска для имени, используемого в определении другу функции (11.3), определенный встроенный в классе предоставления

дружба должна действовать, как описано для поиска в определениях функций члена. Если функция друг

не определен в классе субсидирующего дружбы, поиск имени в определении друга функции поступим

описано для поиска в определениях функций членов пространства имен.

10 В другом объявлении имен функции-члена, имя, используемое в функции описателя и не является частью

Шаблон-аргумент в описатель-ID сначала ищется в рамках класса функция-члена (10.2). Если

он не найден, или если имя является частью шаблона-аргумента в описатель-ID, взгляд на это, как описано

для неквалифицированных имен в определении класса субсидирующего дружбы. [ Пример:

структура А {

ЬурейеЕ Int AT;

недействительными f1 (AT);

недействительными f2 (с плавающей точкой);

шаблон <класс T> недействительным f3 ();

32) Этот поиск применяется ли функция член определяется в пределах определения класса X, или функции-члена

определена в области видимости пространства имен вмещающего определение на Х.

§ 3.4.1

с ISO / IEC 2014 - Все права защищены 45

ISO / IEC 14882: 2014 (E)

};

B {структура

ЬурейеЕ полукокса AT;

TYPEDEF плавать BT;

друг к аннулированию A :: f1 (AT); // Тип параметра A :: AT

друг к аннулированию A :: f2 (BT); // Тип параметра B :: BT

Друг аннулируются A :: f3 <AT> (); // Шаблон аргумент B :: AT

};

-end пример]

11 Во время поиска для имени, используемого в качестве аргумента по умолчанию (8.3.6) в функции параметра декларирование-п

или использовать в экспрессии MEM-инициализатора для конструктора (12.6.2), имена параметров функции,

видимым и скрыть имена лиц, объявленных в блоке, класса или пространства имен областей, содержащих функцию

декларация. [Примечание: 8.3.6 далее описывает ограничения на использование имен в качестве аргументов по умолчанию. 12.6.2

далее описывает ограничения на использование имен в CTOR-инициализатора. -end примечание]

12 Во время поиска имени, используемого в постоянной экспрессии Перечислитель-определения, ранее объявившие

переписчики перечисления видны и скрыть имена лиц, объявленных в блоке, класс, или

пространства имен прицелы, содержащие перечислимую спецификатор.

13 имя, используемое в определении статического члена данных класса X (9.4.2) (после уточненный идентификатор статического

член) ищется, как будто имя было использовано в функции-члене X. [Примечание: 9.4.2 далее описывает

ограничения на использование имен в определении статического элемента данных. -end примечание]

14 Если переменная член пространства имен определяется вне сферы его пространства имен тогда любое имя,

появляется в определении элемента (после описатель-ID) ищется, как будто определения

член произошел в пространстве имен. [ Пример:

Пространство имен {N

Int я = 4;

Экстерн INT J;

}

Int я = 2;

Int N :: J = I; // N :: J == 4

-end пример]

15 имя, используемое в обработчик для функции-Try-блок (раздел 15) ищется, как будто имя было использовано в

внешний блок определения функции. В частности, имена параметров функции не должны быть

повторно объявлен в исключение-декларации, ни во внешнем блоке обработчика для функции попробуйте-блок.

Имена, объявленные в внешнем блоке определения функции не будут найдены, когда ищется в сферу

обработчика для функции попробуйте-блок. [Примечание: Но имена параметров функции найдены. -end примечание]

16 [Примечание: Правила для поиска имен в определениях шаблонов описаны в 14.6. -end примечание]

4 Стандартные преобразования [конв]

1 Стандартные преобразования являются неявные преобразования со встроенным значением. Пункт 4 перечисляет полный набор таких

конверсии. Стандартная последовательность преобразования представляет собой последовательность стандартных преобразований в следующем порядке:

- Ноль или один преобразование из следующего набора: именующее к RValue преобразования массива к указателю преобразования,

и функции к указателю преобразования.

- Ноль или один преобразование из следующего набора: интегральные продвижения по службе, с плавающей точкой продвижение, интеграл

преобразования, с плавающей запятой, преобразования с плавающей интегральные преобразования, преобразования указателей, указатель на

преобразования членов, а также логические преобразования.

- Ноль или один квалификационный преобразования.

[Примечание: стандартная последовательность преобразования может быть пустым, т.е., он может состоять не конверсий. -end примечание]

Стандартная последовательность преобразования будет применен к выражению, если это необходимо, чтобы преобразовать его в требуемое

Тип назначения.

2 [Примечание: выражения с данным типом будут неявно преобразованы в другие типы в нескольких контекстах: