Правительство Российской Федерации
Федеральное государственное автономное образовательное учреждение Высшего образования
Национальный исследовательский университет
«Высшая школа экономики»

Дисциплина «Цифровая грамотность»

Проект
На тему: «Язык программирования C++»

Выполнила:
Новикова Светлана Сергеевна
Студентка 1 курса группы БИЯ-172
Факультет гуманитарных наук, бакалавриат, Иностранные языки и Межкультурная коммуникация

Москва, 2017

Оглавление

1. Общая информация

C++ (читается *си-плюс-плюс*)— компилируемый, статически типизированный язык программирования общего назначения.

Поддерживает такие парадигмы программирования, как процедурное программирование, объектно-ориентированное программирование, обобщённое программирование. Язык имеет богатую стандартную библиотеку, которая включает в себя распространённые контейнеры и алгоритмы, ввод-вывод, регулярные выражения, поддержку многопоточности и другие возможности. C++ сочетает свойства как высокоуровневых, так и низкоуровневых языков. В сравнении с его предшественником — языком C, — наибольшее внимание уделено поддержке объективно-ориентированного и обобщённого программирования

C++ широко используется для разработки программного обеспечения, являясь одним из самых популярных языков программирования. Область его применения включает создания операционных систем разнообразных прикладных программ, драйверов устройств, приложений для встраиваемых систем, высокопроизводительных серверов, а также развлекательных приложений (игр). Существует множество реализаций языка C++, как бесплатных, так и коммерческих и для различных платформ.

Синтаксис C++ унаследован от языка C. Одним из принципов разработки было сохранение совместимости с C. Тем не менее, C++ не является в строгом смысле надмножеством C; множество программ, которые могут одинаково успешно транслироваться как компиляторами C, так и компиляторами C++, довольно велико, но не включает все возможные программы на C.

2. История

Исторический этап развития	*Год*
Язык BCPL	1966
Язык Би (оригинальная разработка Томпсона под UNIX)	1969
Язык Си	1972
Си с классами	1980
C84	1984
Cfront (выпуск E)	1984
Cfront (выпуск 1.0)	1985
Множественное/виртуальное наследование	1988
Обобщённое программирование (шаблоны)	1991
ANSI C++ / ISO-C++	1996
ISO/IEC 14882:1998	1998
ISO/IEC 14882:2003	2003
C++/CLI	2005
TR1	2005
C++11	2011
C++14	2014
C++17	2017

3. Создание

Язык возник в начале 1980-х годов, когда сотрудник фирмы Bell Labs Бьёрн Страуструп придумал ряд усовершенствований к языку C под собственные нужды. [6] Когда в конце 1970-х годов Страуструп начал работать в Bell Labs над задачами теории очередей (в приложении к моделированию телефонных вызовов), он обнаружил, что попытки применения существующих в то время языков моделирования оказываются неэффективными, а применение высокоэффективных машинных языков слишком сложно из-за их ограниченной выразительности. Так, язык Симула имеет такие возможности, которые были бы очень полезны для разработки большого программного обеспечения, но работает слишком медленно, а язык BCPLдостаточно быстр, но слишком близок к языкам низкого уровня и не подходит для разработки большого программного обеспечения.

Вспомнив опыт своей диссертации, Страуструп решил дополнить язык C (преемник BCPL) возможностями, имеющимися в языке Симула. Язык C, будучи базовым языком системы UNIX, на которой работали компьютеры Bell, является быстрым, многофункциональным и переносимым. Страуструп добавил к нему возможность работы с классами и объектами. В результате практические задачи моделирования оказались доступными для решения как с точки зрения времени разработки (благодаря использованию Симула-подобных классов), так и с точки зрения времени вычислений (благодаря быстродействию C). В первую очередь в C были добавлены классы (с инкапсуляцией), наследование классов, строгая проверка типов, inline-функции и аргументы по умолчанию. Ранние версии языка, первоначально именовавшегося «C with classes» («Си с классами»), стали доступны с 1980 года.

Разрабатывая C с классами, Страуструп написал программу cfront[en] — транслятор, перерабатывающий исходный код C с классами в исходный код простого C. Это позволило работать над новым языком и использовать его на практике, применяя уже имеющуюся в UNIX инфраструктуру для разработки на C. Новый язык, неожиданно для автора, приобрёл большую популярность среди коллег и вскоре Страуструп уже не мог лично поддерживать его, отвечая на тысячи вопросов.

К 1983 году в язык были добавлены новые возможности, такие как виртуальные функции, перегрузка функций и операторов, ссылки, константы, пользовательский контроль над управлением свободной памятью, улучшенная проверка типов и новый стиль комментариев (//). Получившийся язык уже перестал быть просто дополненной версией классического C и был переименован из C с классами в «C++». Его первый коммерческий выпуск состоялся в октябре 1985 года.

До начала официальной стандартизации язык развивался в основном силами Страуструпа в ответ на запросы программистского сообщества. Функцию стандартных описаний языка выполняли написанные Страуструпом печатные работы по C++ (описание языка, справочное руководство и так далее). Лишь в 1998 году был ратифицирован международный стандарт языка C++: ISO/IEC 14882:1998 «Standard for the C++ Programming Language»; после принятия технических исправлений к стандарту в 2003 году — следующая версия этого стандарта — ISO/IEC 14882:2003.

4. Развитие и стандартизация языка

В 1985 году вышло первое издание «Языка программирования C++», обеспечивающее первое описание этого языка, что было чрезвычайно важно из-за отсутствия официального стандарта. В 1989 году состоялся выход C++ версии 2.0. Его новые возможности включали множественное наследование, абстрактные классы, статические функции-члены, функции-константы и защищённые члены. В 1990 году вышло «Комментированное справочное руководство по C++», положенное впоследствии в основу стандарта. Последние обновления включали шаблоны, исключения, пространства имён, новые способы приведения типов и булевский тип.

Стандартная библиотека C++ также развивалась вместе с ним. Первым добавлением к стандартной библиотеке C++ стали потоки ввода-вывода, обеспечивающие средства для замены традиционных функций C printf и scanf. Позднее самым значительным развитием стандартной библиотеки стало включение в неё Стандартной библиотеки шаблонов.

В 1998 году был опубликован стандарт языка ISO/IEC 14882:1998 (известный как C++98),[8] разработанный комитетом по стандартизации C++ (ISO/IEC JTC1/SC22/WG21 working group). Стандарт C++ не описывает способы именования объектов, некоторые детали обработки исключений и другие возможности, связанные с деталями реализации, что делает несовместимым объектный код, созданный различными компиляторами. Однако для этого третьими лицами создано множество стандартов для конкретных архитектур и операционных систем.

В 2003 году был опубликован стандарт языка ISO/IEC 14882:2003, где были исправлены выявленные ошибки и недочёты предыдущей версии стандарта.

В 2005 году был выпущен отчёт Library Technical Report 1 (кратко называемый TR1). Не являясь официально частью стандарта, отчёт описывает расширения стандартной библиотеки, которые, как ожидалось авторами, должны быть включены в следующую версию языка C++. Степень поддержки TR1 улучшается почти во всех поддерживаемых компиляторах языка C++.

С 2009 года велась работа по обновлению предыдущего стандарта, предварительной версией нового стандарта сперва был C++09, а спустя год C++0x, сегодня[когда?] — C++11, куда были включены дополнения в ядро языка и расширение стандартной библиотеки, в том числе большую часть TR1.

C++ продолжает развиваться, чтобы отвечать современным требованиям. Одна из групп, разрабатывающих язык C++ и направляющих комитету по стандартизации C++ предложения по его улучшению — это Boost, которая занимается, в том числе, совершенствованием возможностей языка путём добавления в него особенностей метапрограммирования.

Никто не обладает правами на язык C++, он является свободным. Однако сам документ стандарта языка (за исключением черновиков) не доступен бесплатно. В рамках процесса стандартизации, ISO выпускает несколько видов изданий. В частности, технические доклады и технические характеристики публикуются, когда «видно будущее, но нет немедленной возможности соглашения для публикации международного стандарта.» До 2011 года не было опубликовано три технических отчёта по C++: TR 19768: 2007 (также известный как C++, Технический отчёт 1) для расширений библиотеки в основном интегрирован в C++11, TR 29124: 2010 для специальных математических функций, и TR 24733: 2011 для десятичной арифметики с плавающей точкой.

5. История названия

Имя языка, получившееся в итоге, происходит от оператора унарного постфиксного инкремента C ++ (увеличение значения переменной на единицу). Имя C+ не было использовано потому, что является синтаксической ошибкой в C и, кроме того, это имя было занято другим языком. Язык также не был назван D, поскольку «является расширением C и не пытается устранять проблемы путём удаления элементов C».

6. Философия C++

В книге «Дизайн и эволюция C++» [11] Бьёрн Страуструп описывает принципы, которых он придерживался при проектировании C++. Эти принципы объясняют, почему C++ именно такой, какой он есть. Некоторые из них:

- Получить универсальный язык со статическими типами данных, эффективностью и переносимостью языка C.
- Непосредственно и всесторонне поддерживать множество стилей программирования, в том числе процедурное программирование, абстракцию данных, объектно-ориентированное программирование и обобщённое программирование.
- Дать программисту свободу выбора, даже если это даст ему возможность выбирать неправильно.
- Максимально сохранить совместимость с C, тем самым делая возможным лёгкий переход от программирования на C.
- Избежать разночтений между C и C++: любая конструкция, допустимая в обоих языках, должна в каждом из них обозначать одно и то же и приводить к одному и тому же поведению программы.
- Избегать особенностей, которые зависят от платформы или не являются универсальными.
- «Не платить за то, что не используется» — никакое языковое средство не должно приводить к снижению производительности программ, не использующих его.
- Не требовать слишком усложнённой среды программирования.

7. Специальные функции

Класс по умолчанию может иметь шесть специальных функций: конструктор по умолчанию, конструктор копирования, конструктор перемещения, деструктор, оператор присваивания копированием, оператор присваивания перемещением. Также можно явно определить их все.

```cpp
class Array {
public:
    Array() = default; // компилятор создаст конструктор по-умолчанию сам
    Array(size_t _len) :
            len(_len) {
        val = new double[_len];
    }
    Array(const Array & a) = delete;  // конструктор копирования явно удалён
    Array(Array && a); // конструктор перемещения
    ~Array() {
        delete[] val;
    }
    Array& operator=(const Array& rhs); // оператор присваивания копированием
    Array& operator=(Array&& rhs); // оператор присваивания перемещением
    double& operator[](size_t i) {
        return val[i];
    }
    const double& operator[](size_t i) const {
        return val[i];
    }

protected:
    std::size_t len {0}; // инициализация поля
    double* val {nullptr};
};
```

Рисунок 1 (Функция C++)

Конструктор вызывается для инициализации объекта (соответствующего типа) при его создании, а деструктор — для уничтожения объекта. Класс может иметь несколько конструкторов, но деструктор может иметь только один. Конструкторы в C++ не могут быть объявлены виртуальными, а деструкторы — могут, и обычно объявляются для всех полиморфных типов, чтобы гарантировать правильное уничтожение доступного по ссылке или указателю объекта независимо от того, какого типа ссылка или указатель. При наличии хотя бы у одного из базовых классов виртуального деструктора, деструктор класса потомка автоматически становится виртуальным.