Дата последней сборки документа:
24 августа 2023 г. в 00:36

Приемы программирования на языке C

Содержание

1. Ресуры по языку C

https://learnc.info/c/

2. Visual Studio Code как среда разработки для языка C

Скачать Visual Studio Code можно здесь https://code.visualstudio.com/. Для ОС Windows нужно еще установить GCC. На ОС Linux компилятор gcc доступен «из коробки». На ОС MacOS компилятор gcc можно установить с помщью утилиты `brew`.

После установки IDE останется только создать директорию с проектом под язык Си. Когда Visual Studio Code увидит файл с расширением `*.c`, она предложит установить специальное расширение «C/C++ Extension Pack v1.X.X».

3. Алфавит, идентификаторы, служебные слова

Идентификаторы, начинающиеся с одного символа подчеркивания «_» или с двух символов подчеркивания «__», зарезервированы для использования в библиотеках и компиляторах. Поэтому такие идентификаторы не рекомендуется выбирать в качестве имен в прикладной программе на языке Си. Рекомендуется при программировании имена констант записывать целиком заглавными буквами [2, стр. 15].

3.1. Константы и строки

По определению, константа представляет значение, которое не может быть изменено. Синтаксис языка определяет 5 типов *констант*:

1. символы,
2. константы перечисляемого типа,

3. вещественные числа,
4. целые числа,
5. нулевой указатель («null»-указатель).

Управляющие последовательности (`'\n'`, `'\r'`, etc.) являются частным случаем экскейп-последовательнос (ESC-последовательностей), к которым также относятся лексемы вида `'\ddd'`, либо `'\xhh'`.

Символьная константа (символ) имеет *целый тип*, то есть символы можно использовать в качестве целочисленных операндов в выражениях.

Целочисленные именованные константы можно вводить с помощью перечисления `enum`. Пример

```
enum DAY {SUNDAY, MONDAY, ...};
enum BOOLEAN {NO, YES};
```

В первой строке `DAY`, а во второй строке `BOOLEAN` это необязательный произвольный идентификатор – название перечисления.

Если в списке нет ни одного элемента со знаком `'='`, то значения констант начинаются с 0 и увеличиваются на 1 слева направо. Таким образом, `NO` равно 0, а `YES` – 1. Именованная константа со знаком `'='` получает соответствующее значение, а следующая за ней именованные константы без явных значений увеличиваются на 1 каждая.

То есть если

```
enum BOOLEAN {NO=10, YES};
printf("NO=%d, YES=%d", NO, YES);  // NO=10, YES=11
```

В Python можно сделать так

```
from enum import Enum, auto

class Boolean(Enum):
    NO = 0
    YES = auto()

Boolean.NO.value  # 0
Boolean.YES.value  # 1
```

Формально строки не относятся к константам языка Си, а представляют собой отдельный тип его лексем. Строковая константа определяется как последовательность символов, заключенных в двойные кавычки (не в апострофы).

Представление *строковых констант* в памяти ЭВМ подчиняются следующим правилам. Все символы строки размещаются подряд, и каждый символ (в том числе представленный эскейп-последовательностью) занимает ровно 1 байт. В конце записи строковой константы компилятор помещает символ `'\0'`.

Таким образом, количество байтов, выделяемое в памяти ЭВМ для представления значения строки, ровно на 1 больше, чем число символов в записи этой строковой константы.

При работе с символьной информацией нужно помнить, что длина символьной константы `'F'` равна 1 байту, а длина строки `"F"` равна 2 байтам.

3.2. Переменные и именованные константы

Одним из основных понятий языка Си является *объект* – именованная область памяти. Частный случай объекта – переменная.

Каждый из целочисленных типов (`char`, `short`, `int`, `long`) может быть определен либо как *знаковый* `signed` либо как *беззнаковый* `unsigned` (по умолчанию `signed`).

Различие между этими двумя типами – в правилах интерпретации *старшего бита внутреннего представления*. Спецификатор `signed` означает, что старший бит внутреннего представления воспринимался как знаковый; `unsigned` означает, что старший бит внутренного представления входит в код представляемого числового значения, которое считается в этом случае беззнаковым. Выбор знакового или беззнакового представления определяет предельные значения, которые можно представить с помощью описанной переменной. Например на IBM PC переменная типа `unsigned int` позволяет представить числа от 0 до 65 535, а переменная типа `signed␣int` (или просто `int`) соответствуют значения в диапазоне от -32768 до +32767.

Именованные константы можно вводить с помощью *директивы препроцессора* `#define`, напрмер

```
#define EULER 2.718282  // точка с запятой не нужна!!!
```

Что эквивалентно

```
const double EULER = 2.718282;
```

До начала компиляции текст программы на языке Си обрабатывается специальным компонентом транслятора – *препроцессором*. Далее текст от препроцессора поступает к компилятору. Итак, основное отличие констант, определяемых препроцессорными директивами `#define`, состоит в том, что эти *константы вводятся* в текст программы *до этапа компиляции* – препроцессор обрабатывает исходный код программы и делает в этом тексте замены и подстановки [2, стр. 29].

Список литературы

1. *Кольцов Д.М.* Си на примерах. Практика, практика и только практика. – СПб.: Наука и Техника, 2019. – 288 с.
2. *Подбельский В.В.*, *Фомин С.С.* Прграммирование на языке Си, 2005. – 600 с.

Листинги