C

Встроенные типы данных

Название	Обозначение	Диапазон значений
Пустота	void	
Байт	char	от -128 до +127
без знака	unsigned char	от 0 до 255
Короткое целое число	short	от -32768 до +32767
Короткое целое число без знака	unsigned short	от 0 до 65535

Встроенные типы данных

Название	Обозначение	Диапазон значений
Целое число	int	от – 2147483648 до + 2147483647
Целое число без знака	unsigned int (или просто unsigned)	от 0 до 4294967295
Длинное целое число	long	от – 2147483648 до + 2147483647
Длинное целое число без знака	unsigned long	от 0 до 4294967295

Встроенные типы данных

Название	Обозначение	Диапазон значений
Вещественное число одинарной точности	float	от ±3.4e-38 до ±3.4e+38 (7 значащих цифр)
Вещественное число двойной точности	double	от ±1.7e-308 до ±1.7e+308 (15 значащих цифр)
Вещественное число увеличенной точности	long double	от ±1.2e-4932 до ±1.2e+4932

Производные типы данных

- ***Указатель*** – это производный тип, который представляет собой *адрес* какого-либо значения.
- Тип*

Производные типы данных

- ***Массив*** – это *коллекция* нескольких величин одного и того же типа.
- Тип имя[число величин]
- Тип имя[число величин 1] [число величин 2]
- Тип имя[число величин 1] [число величин 2]… [число величин n]
- Доступ: имя[индекс]; имя[индекс 1][индекс 2]; имя[индекс 1][индекс 2]… [индекс n];

Производные типы данных

- *Строки* представляются в виде *массива* байтов: char имя[размер строки];
- Для записи строковых *констант* в программе используются *литералы*.
- ***Литерал*** – это последовательность знаков, заключенная в двойные кавычки:
- "Это строка" "0123456789" "*"

Производные типы данных

- Перечисляемый тип - тип данных, чьё множество значений представляет собой ограниченный список идентификаторов.
- enum имя{идентификатор, …}
- enum имя{идентификатор = значение, …}
- enum DayTime { morning, day, evening, night };

Производные типы данных

- Структура представляет собой переменную, группирующую связанные части информации, называемые элементами, типы которых могут различаться.
- struct имя
- {
- Тип «имя поля»; …
- }

Операция sizeof

- Операция sizeof в качестве аргумента берет имя типа или *выражение*. Аргумент заключается в скобки (если аргумент – *выражение*, скобки не обязательны). Результат операции – целое число, равное количеству байтов, которое необходимо для хранения в памяти заданной величины.
- sizeof(long);
- // сколько байтов занимает тип long
- sizeof (b);
- // сколько байтов занимает переменная b

Тернарная операция

- ***Тернарная операция*** ; если значение первого операнда – истина, то результат – второй операнд; если ложь – результат – третий операнд. Первый операнд должен быть логическим значением, второй и третий операнды могут быть любого, но одного и того же типа, а результат будет того же типа, что и третий операнд.
- операнд1?операнд2:операнд3

Бинарные операции

Арифметические операции

- + сложение
- - вычитание
- * умножение
- / деление
- % остаток

Операции сравнения

- == равно
- != не равно
- < меньше
- > больше
- <= меньше или равно
- >= больше или равно

Бинарные операции

Логические операции

- && логическое И
- || логическое ИЛИ
- ! логическое НЕ

Битовые операции

- & битовое И
- | битовое ИЛИ
- ^ битовое ИСКЛЮЧАЮЩЕЕ ИЛИ
- ~ битовое НЕ
- << сдвиг влево
- >> сдвиг вправо

Бинарные операции

Операции присваивания

- = присваивание
- +=, -=, *=, /=, %=, |=, &=, ^=, <<=, >>= выполнить операцию и присвоить

Последовательность

- , последовательность
- Выполнить *выражение* до запятой,
  затем *выражение* после запятой. Два произвольных *выражения* можно поставить рядом, разделив их запятой. Они будут выполняться **последовательно**, и результатом всего *выражения* будет результат последнего *выражения*.

Унарные опеации

Арифметические операции

- ++ увеличить на единицу, префиксная и
- постфиксная формы
- -- уменьшить на единицу, префиксная и
- постфиксная формы

Прочие

- & взятие *адреса*
- * обращение по адресу
- [] индекс массива
- () вызов функции
- . доступ к полю

Операторы

Ветвления

- If (условие)
- Выражение 1;
- else
- Выражение 2;

- switch (выражение)
- {
- case константа1: выражение
- break;
- default: выражение
- }

Циклы

- for(тип индекс = нач.знач.; условие; изменение индекса) тело цикла;
- while (условие) тело цикла;
- do {тело цикла} while(условие);

Операторы

- continue - обеспечивает передачу управления управляющему выражению наименьшего внешнего цикла.
- break - заканчивает выполнение ближайшего внешнего цикла или условного оператора, в котором он отображается.

Функции

- Тип «имя функции»(«тип 1» «параметр 1», «тип 2» «параметр 2», …, «тип n» «параметр n», «тип n + 1» «параметр n + 1» = «значение 1», …, «тип n + m» «параметр n + m» = «значение m»)
- {

  return «значение функции»;
- };

Функции

- int main(int argc, char* argv[])
- {
- return 0;
- }
- main – точка входа в программу

Функции

- #include <stdlib.h>
- void *malloc(size_t *количество_байтов*);
- выделяет память
- void free(void **p*)
- освобождает участок памяти
- Int *a;
- a = malloc(40*sizeof(int));
- int (*p)[10];
- p = malloc(40*sizeof(int));

Функции

- #include <stdio.h>
- int scanf(char *управляющая строка);
- int printf(char *управляющая строка, ...);

Функции

- **Спецификаторы формата:**
- %c символ
- %d целое десятичное число
- %i целое десятичное число
- %e десятичное число в виде x.xx e+xx
- %E десятичное число в виде x.xx E+xx
- %f десятичное число с плавающей запятой xx.xxxx
- %F десятичное число с плавающей запятой xx.xxxx
- %g %f или %e, что короче
- %G %F или %E, что короче
- %o восьмеричное число
- %s строка символов
- %u беззнаковое десятичное число
- %x шестнадцатеричное число
- %X шестнадцатеричное число
- %% символ %
- %p указатель
- %n указатель
- %ld печать long int
- %hu печать short unsigned
- %Lf печать long double
- \b BS, забой
- \f Новая страница, перевод страницы
- \n Новая строка, перевод строки
- \r Возврат каретки
- \t Горизонтальная табуляция
- \v Вертикальная табуляция
- \« Двойная кавычка
- \‘ Апостроф
- \\ Обратная косая черта
- \0 Нулевой символ, нулевой байт
- \a Сигнал
- \N Восьмеричная константа
- \xN Шестнадцатеричная константа
- \? Знак вопроса

Функции

- int **rand** (void);
- генерирует псевдослучайное значение
- void **srand** (unsigned int seed);
- устанавливает начальное значение генератора псевдослучайных чисел
- RAND_MAX – максимальное значение.

Работа с файлами

- #include <stdio.h>
- FILE * fo;
  fo = fopen("test.txt","wt");

- Можно задать и полный путь к файлу, например:
- fo = fopen("c:\\tmp\\test.txt","wt");
- if ((fo=fopen("c:\\tmp\\test.txt","wt")) == 0)
  // ошибка!
- fclose(fo);

Работа с файлами

- fprintf(fo, "Вывод: %s %d", str, n);
- fscanf(fi, "%d", &n);

Работа с файлами

- #include <stdio.h>
- FILE * **fopen** (const char * filename, const char * mode);
- FILE * fopen(имя физического файла, режим доступа)
- int fclose (FILE * stream);
- int **fprintf** (FILE * stream, const char * format, ...);
- int fscanf (FILE * stream, const char * format, ...);

Работа с файлами

Значение	Описание
r	Файл открывается только для чтения
w	Файл открывается только для записи. Если соответствующий физический файл существует, он будет перезаписан
a	Файл открывается для записи в конец (для дозаписи) или создается, если не существует
r+	Файл открывается для чтения и записи.
w+	Файл открывается для записи и чтения. Если соответствующий физический файл существует, он будет перезаписан
a+	Файл открывается для записи в конец (для дозаписи) или создается, если не существует