СТАНДАРТНАЯ БИБЛИОТЕКА СИ

Тема 1. Раздел 1. Вводная лекция

План

- История
- Задачи
- Структура
- Библиотека ввода\вывода
- fprintf и fscanf

История

- Язык программирования Си до стандартизации не обеспечивал встроенной функциональности, как, например, операции ввода-вывода
- В 1983 году Американский национальный институт стандартов (ANSI) сформировал комитет для принятия стандарта языка Си, известный как «ANSI Си»
- Эта работа вылилась в создание так называемого стандарта C89 в 1989. Часть итогового стандарта была набором библиотек, названная **Стандартная библиотека ANSI Си**.
- Последующие версии стандарта языка Си добавляли некоторые новые и наиболее полезные заголовочные файлы в библиотеку. Поддержка этих новых расширений зависела от реализации.

Задачи библиотеки

- основной набор математических функций
- обработка строк
- конвертация типов
- файловый и консольный ввод-вывод

Структура

- Стандартная библиотека ANSI Си состоит из 24 заголовочных файлов
- Каждый из которых можно подключать к программному проекту при помощи одной директивы
- Каждый заголовочный файл содержит объявления одной или более функций, определения типов данных и макросы.

Список заголовочных файлов

Имя	Описание
<assert.h>	Содержит макрос утверждений, используемый для обнаружения логических и некоторых других типов ошибок в отлаживаемой версии программы
<complex.h>	Набор функций для работы с комплексными числами.
<ctype.h>	Содержит функции, используемые для классификации символов по их типам или для конвертации между верхним и нижним регистрами независимо от используемой кодировки
<errno.h>	Для проверки кодов ошибок, возвращаемых библиотечными функциями.
<fenv.h>	Для управления средой, использующей числа с плавающей запятой.
<float.h>	Содержит заранее определенные константы, описывающие специфику реализации свойств библиотеки для работы с числами с плавающей запятой
<inttypes.h>	Для точной конвертации целых типов.

Список заголовочных файлов

Имя файла	Описание
<iso646.h>	Для программирования в кодировке ISO 646.
<limits.h>	Содержит заранее заданные константы, определяющие специфику реализации свойств целых типов, как, например, область допустимых значений (_MIN, _MAX).
<locale.h>	Для setlocale() и связанных констант. Используется для выбора соответствующего языка.
<math.h>	Для вычисления основных математических функций
<setjmp.h>	Объявляет макросы setjmp и longjmp, используемые для переходов
<signal.h>	Для управления различными исключительными условиями
<stdarg.h>	Для доступа к различному числу аргументов, переданных функциям.

Список заголовочных файлов

Имя файла	Описание
<stdbool.h>	Для булевых типов данных.
<stdint.h>	Для определения различных типов целых чисел.
<stddef.h>	Для определения нескольких полезных типов и макросов.
<stdio.h>	Реализует основные возможности ввода и вывода в языке Си.
<stdlib.h>	Для выполнения множества операций, включая конвертацию, генерацию псевдослучайных чисел, выделение памяти, контроль процессов, окружения, сигналов, поиска и сортировки.
<string.h>	Для работы с различными видами строк.
<tgmath.h>	Для типовых математических функций.

Список заголовочных файлов

Имя файла	Описание
<time.h>	Для конвертации между различными форматами времени и даты.
<wchar.h>	Unicode
<wctype.h>	Unicode

Stdio.h

Библиотека ввода-вывода

Stdio.h. Задачи

- Работа с файлами
- Форматированный ввод-вывод
- Вывод ошибок

Stdio.h. Функция Printf

Print formatted output to the standard output stream

int printf(const char *format [, argument]...);

Return Value.

Returns the number of characters printed, or a negative value if an error occurs.

Stdio.h. Функция Scanf

Read formatted data from the standard input stream

int scanf(const char *format [, argument]...);

Return Value

Returns the number of fields successfully converted and assigned; the return value does not include fields that were read but not assigned. A return value of 0 indicates that no fields were assigned.

Stdio.h. Ввод-вывод. Example

```
int _tmain(int argc, _TCHAR* argv[])
{
    printf("Hello World!\n ");

    char a;
    scanf(&a);

    return 0;
}
```

Формат ввода-вывода

%[flags] [width] [.precision] type

Type

тип вводимого параметра

Flags

специфика формата выводимого значения

Width

Минимальное выводимое число символов

Precision

Максимальное выводимое число символов после запятой

Формат ввода выводы. Type

Символ	Тип	Формат
C or c	int	When used with **printf** functions, specifies a single-byte character
d or i	int	Signed decimal integer
o	int	Unsigned octal integer
u	int	Unsigned decimal integer
x	int	Unsigned hexadecimal integer, using "abcdef.”
X	int	Unsigned hexadecimal integer, using "ABCDEF."
e	double	Signed value having the form [–]d**.**dddd **e** [*sign*]dd[d] where d is a single decimal digit, dddd is one or more decimal digits, dd[d] is two or three decimal digits depending on the output format and size of the exponent, and *sign* is + or –
E	double	Identical to the **e** format except that **E** rather than **e** introduces the exponent
f	double	Signed value having the form [–]dddd**.**dddd, where dddd is one or more decimal digits. The number of digits before the decimal point depends on the magnitude of the number, and the number of digits after the decimal point depends on the requested precision.

Формат ввода-вывод. Type

Символ	Тип	Описание
g	double	Signed value printed in **f** or **e** format, whichever is more compact for the given value and precision. The **e** format is used only when the exponent of the value is less than –4 or greater than or equal to the precision argument. Trailing zeros are truncated, and the decimal point appears only if one or more digits follow it.
G	double	Identical to the **g** format, except that **E**, rather than **e**, introduces the exponent (where appropriate).
A or a	double	Signed hexadecimal double precision floating point value having the form [−]0x*h.hhhh* **p±**dd, where *h.hhhh* are the hex digits (using lower case letters) of the mantissa, and dd are one or more digits for the exponent. The precision specifies the number of digits after the point.
n	Point to integer	Number of characters successfully written so far to the stream or buffer; this value is stored in the integer whose address is given as the argument. See Security Note below
p	Pointer to void	Prints the argument as an address in hexadecimal digits
S or s	String	When used with **printf** functions, specifies a single-byte–character string

Ввод-вывод

- int puts(const char *str)
- int putc(int c)

- char *gets(char* buffer);
- int getchar();

Question Time