Best practices программирования на C

Документы

- SEI CERT C Coding Standard
- MISRA C
- NASA C style guide, Mars Code
- Micriμm C Coding Standard
- Steve Oualline. C Elements of Style: The Programmer's Style Manual for Elegant C and C++ Programs
- Марсокод, или Как создавалось ПО для марсохода Curiosity
- 47 Атрибутов Хорошего С-кода
- **Worst practice**: "С с классами". Orthodox C++. Modern C for C++ Peeps.

Style guides

- Linux kernel coding style
  - Kernel Coding the Upstream Way - Tim Bird, Sony
- OpenSSL coding style
- FreeBSD kernel style guide
- GNU coding standards
- Apache Developers' C Language Style Guide
- OpenResty C Coding Style Guide, ngx-releng

Автоматические форматтеры кода

- GNU indent
- Uncrustify
  - UncrustifyConfig
- clang-format
  - clang-format configurator

Best practices

Избегаем UB

MISRA Rule 1.2: No reliance shall be placed on undefined or unspecified behaviour.

- целочисленные переполнения

```
int64_t bill_gates_fortune;
for(size_t i = 0; i < strlen(str); i++);
```

- неинициализированные переменные

```
int i;
printf("%d\n", i);

char* str = malloc(4096);
printf("%s\n", str);
```

Debian and OpenSSL: The Aftermath, Why is uninitialized memory safe to use in OpenSSL's random number generator? - Stack Overflow

- ошибки доступа к памяти

  - доступ к уже удалённой через `free` динамической памяти
  - доступ к переменной на стеке после возврата из родительской функции
  - доступ за пределами памяти
    * Stack buffer overflow - Wikipedia
    * 2021 CWE Top 25 Most Dangerous Software Weaknesses
    * Использование подсказок, включаемых в исходный код, помогающих GCC выявлять случаи переполнения буфера
  - утечка памяти
    * Pointers and memory leaks in C

Bounds-checking interfaces

- ISO/IEC TR 24731-1, C11 Annex K

```
// #ifdef __STDC_LIB_EXT1__
// #define __STDC_WANT_LIB_EXT1__
errno_t strcpy_s(char* restrict s1,
                 rsize_t s1max,
                 const char* restrict s2);
```

> This function or variable may be unsafe. Consider using safe-version instead. To disable deprecation, use _CRT_SECURE_NO_WARNINGS. See online help for details.

– MSVC

Security Features in the CRT | Microsoft Docs

- `strncpy`, `strncat`. Безопасность
- `strlcpy`, `strlcat` (BSD, source)
- `strnlen`
- `strerror_r`
- `strtok_r`
- `fgets`
- `ctime_r`, `gmtime_r`, `asctime_r`, `localtime_r`
- `snprintf`, `vsnprintf`
- `asprintf`, `vasprintf`
- Обнаружено еще одно имя сети Wi-Fi, выводящее из строя iPhone

```
// Bad
printf(user_str);
// Good
printf("%s", user_str);
// Good
puts(user_str);
```

Статические анализаторы кода

- Lint
- Clang Static Analyzer, clang-tidy
- Cppcheck
- PVS studio
  - блог на Хабре
- Intel ControlFlag

John Carmack. In-Depth: Static Code Analysis

Тестирование СУБД: 10 лет опыта

Compiler Sanitizers

- Compiler Sanitizers - HPC Wiki
- Undefined Behaviour Sanitizer

  Instrumentation Options (Using GCC)

```
$ cat test.c
int main(int argc, char **argv) {
    int k = 0x7fffffff;
    k += argc;
    return 0;
}
$ gcc -fsanitize=undefined test.c
$ ./a.out
test.c:3:5: runtime error: signed integer overflow: 2147483647 + 1 cannot
be represented in type 'int'
```

- Address Sanitizer

  AddressSanitizer · google/sanitizers Wiki

```
$ cat test.c
#include <stdlib.h>
int main() {
  char *x = malloc(10 * sizeof(char));
  free(x);
  return x[5];
}
$ gcc -fsanitize=address test.c
$ ./a.out
==25480==ERROR: AddressSanitizer: heap-use-after-free on address 0x60700000dfb5 at
pc 0x0000004007d4 bp 0x7fff7645f390 sp 0x7fff7645f380
```

Defensive programming

- Defensive programming - Wikipedia
- assert

```
int process_string(char* str)
{
    assert(str != NULL);
    // ...
}

// Bad
assert(str && str[0] && strlen(str) > nSize);
// Good
assert(str != NULL);
assert(str[0] != 0);
assert(strlen(str) > nSize);

// Bad
assert(0);
// Good
assert(!"This should never happen, contact devs");

// Bad
assert(str != NULL);
puts(str);

// Good
assert(str != NULL);
if(str == NULL)
```

```
        return NULL;
    puts(str);
```

- проверка возвращаемых значений (`-Wunused-result`, `FORTIFY_SOURCE`)

```
char* str = malloc(4096);
if(!str)
{
    perror("malloc");
    exit(EXIT_FAILURE);
}
```

- `FORTIFY_SOURCE` 1/2 в GCC старше 4.0. Enhance application security with FORTIFY_SOURCE, перевод

```
gcc source.c -O1 -D_FORTIFY_SOURCE=2
```

- журналирование aka logging

Именование

```
// Константы, макросы
#define MY_CONSTANT 42
#define MY_MACRO(x) ((x)*(x))

// Элементы перечислений
enum color
{
    RED,
    GREEN,
    BLUE
};

// Типы
typedef struct
{
    int value;
} data;
data d;

// Переменные
int my_var;
for(int i = 0; i < size; i++);

// Функции
// module-object-operation
int os_semaphore_post(void);
```

Венгерская нотация - Википедия

```
wc.lpfnWndProc   = WindowProc;
wc.hInstance     = hInstance;
wc.lpszClassName = CLASS_NAME;
```

> Этот стиль выбора имён называется «венгерской» записью по названию родины руководителя отдела программирования Microsoft Чарльза Симони, который его изобрёл. (А не потому, что его использование придаёт программам такой вид, будто они написаны на венгерском языке).

– А. Голуб. Верёвка достаточной длины, чтобы выстрелить себе в ногу

Лимиты

- C17 5.2.4.1:
  - 63 significant initial characters in an internal identifier or a macro name
  - 31 significant initial characters in an external identifier
  - 511 identifiers with block scope declared in one block
  - 4095 external identifiers in one translation unit

  - 127 nesting levels of blocks
  - 127 parameters in one function/macro
  - 4095 characters in a logical source line
  - 4095 characters in a string literal
  - 1023 members in a single structure or union
  - 1023 enumeration constants in a single enumeration

SEI CERT DCL40-C

Зарезервированные имена SEI CERT DCL37-C

- Reserved Names (The GNU C Library)

  The names of all library types, macros, variables and functions that come from the ISO C standard are reserved unconditionally; your program may not redefine these names. All other library names are reserved if your program explicitly includes the header file that defines or declares them.

  reserved names include all external identifiers (global functions and variables) that begin with an underscore ('_') and all identifiers regardless of use that begin with either two underscores or an underscore followed by a capital letter

Комментарии

- Code Tells You How, Comments Tell You Why
- `#if 0`
- `#endif  // ...`

Декомпозиция и возможности языка

C Interfaces and Implementations

- `*.c` + `*.h` = "модуль"
  - Google C++ Style Guide: Names and Order of Includes
- "include guards":

```
#ifndef MYFILE_H
#define MYFILE_H
// ...
#endif  // MYFILE_H
```

- сокрытие: `extern` vs `static`, opaque pointer
- `const`

```
char* const a;  // pointer is constant, the data is not
const char* a;  // pointed data cannot be written using pointer
```

- modern features: `bool`, VLA, designated initializers, generics, compound literals etc.

```
// Bad
int v = 1;
setsockopt( fd, SOL_SOCKET, SO_REUSEADDR, &v, sizeof v );

// Good
setsockopt( fd, SOL_SOCKET, SO_REUSEADDR, &( int ){ 1 }, sizeof int );
```

- GOTOphobia considered harmful (in C), О вреде GOTO-фобии (с примерами на C)