

## Задание 1

### Необходимые знания

1. Как менять значения переменных внутри функций в Си.

### Задание

В папке swap лежит 3 файла: swap.c, swap.h и main.c. Ваша задача закончить метод Swap в swap.c, так, чтобы он менял местами два символа. Скомпилировать программу. Если вы все сделали верно, то программа, которую вы собрали выведет "b a".

## Изменение значений переменных внутри функций в C

### 1. Передача по значению

Функция получает **копию** переменной. Изменения не сохраняются после вызова.

### 2. Передача по указателю

Функция получает **адрес** переменной. Изменения сохраняются.

Код поменял, все закоммитил

```
@ccurecc →.../os_lab_2019/lab2/src/swap (master) $ gcc swap.c main.c -o swap_program
@ccurecc →.../os_lab_2019/lab2/src/swap (master) $ ./swap_program
b a
@ccurecc →.../os_lab_2019/lab2/src/swap (master) $
```

## Задание 2

### Необходимые знания

1. Выделение и освобождение памяти в куче Си.
2. В чем разница между стеком и кучей (прямо в задании не потребуется, но я спрошу).
3. Использование аргументов командной строки

### Задание

В папке revert\_string содержатся файлы main.c, revert\_string.h, revert\_string.c. Вам необходимо реализовать метод RevertString в revert\_string.c, который должен переворачивать данную пользователем строку. Изучить код main.c, скомпилировать программу, рассказать, как она работает и, что делает.

## 1. Выделение и освобождение памяти в куче (Си)

### Выделение:

- **malloc(size)** — выделяет size байт неинициализированной памяти.
- **calloc(n, size)** — выделяет память для n элементов по size байт (инициализирует нулями).
- **realloc(ptr, new\_size)** — изменяет размер ранее выделенного блока памяти.

## Освобождение:

- **free(ptr)** — освобождает память, выделенную malloc/calloc/realloc.

## 2. Разница между стеком и кучей

Стек	Куча
Автоматическое управление памятью	Ручное управление (malloc/free)
Быстрый доступ	Медленнее стека
Ограниченный размер	Большой размер (доступна вся RAM)
Хранит локальные переменные	Хранит динамические данные
Освобождается при выходе из функции	Требуется явное освобождение

## 3. Аргументы командной строки

Передача параметров в программу при запуске.

```
int main(int    , char *    []) {  
    // argc — количество аргументов (включая имя программы)  
    // argv[] — массив строк (аргументы)  
}
```

Пример:

```
#include <stdio.h>
```

```
int main(int    , char *    []) {  
    printf("Программа: %s\n",    [0]);  
    for (int = 1; <    ; ++){  
        printf("Аргумент %d: %s\n", ,    [ ]);  
    }  
}
```

Запуск:

```
./program hello 123
```

```
# Выведет:  
# Программа: ./program  
# Аргумент 1: hello  
# Аргумент 2: 123
```

Код поменял, все закоммитил, код мейна изучил, написал подробные комментарии

```
● @ccurecc →.../os_lab_2019/lab2/src/revert_string (master) $ gcc main.c revert_string.c -o revert_string  
● @ccurecc →.../os_lab_2019/lab2/src/revert_string (master) $ ./revert_string "Venom"  
  Reverted: noneV  
○ @ccurecc →.../os_lab_2019/lab2/src/revert_string (master) $
```

## Задание 3

### 🔗 Необходимые знания

1. Основы работы компилятора: препроцессор, компилятор, линковщик. Их роли и порядок работы.
2. Что такое статическая и динамическая линковка. В чем разница?
3. Опции компилятора: `-I`, `-L`, `-l`, `-shared`, `-o`, `-c`, `-fPIC`
4. Утилита `ar`.
5. Переменная окружения `LD_LIBRARY_PATH`

### Задание

В задании 2, вы написали маленькую библиотеку с одной функцией, переворота строки. Тем не менее этот код уже можно переиспользовать, а чтобы это было удобнее делать, его необходимо вынести в библиотеку. Ваше задание скомпилировать статическую и динамическую библиотеки с `RevertString` и залинковать их в приложения с `main.c`.

*Получится две программы, первая будет использовать статическую библиотеку, а вторая динамическую.*

### 1. Этапы работы компилятора

#### 1. Препроцессор (`gcc -E`)

Обрабатывает директивы `#include`, `#define`, удаляет комментарии.

**На выходе:** чистый C-код (`.i` файл).

#### 2. Компилятор (`gcc -S`)

Преобразует код в ассемблер (`.s` файл).

#### 3. Ассемблер (`gcc -c`)

Генерирует объектный файл (`.o`).

#### 4. Линковщик (`ld`)

Связывает объектные файлы и библиотеки в исполняемый файл.

### 2. Статическая vs динамическая линковка

Статическая

Динамическая

Библиотеки (`.a`) вшиваются в бинарник

Библиотеки (`.so`) подгружаются при запуске

Большой размер файла

Меньший размер

Не зависит от системы

Требует совместимых `.so` на целевой системе

Изменения в `.so` не влияют

Обновления `.so` применяются автоматически

### 3. Опции компилятора

Опция Назначение Пример

-I	Добавить путь к заголовочным файлам	gcc -I/usr/local/include main.c
-L	Добавить путь к библиотекам	gcc -L/usr/local/lib -lm
-l	Связать библиотеку	gcc -lpthread main.c
-shared	Создать динамическую библиотеку	gcc -shared -o libfoo.so foo.c
-o	Задать имя выходного файла	gcc main.c -o program
-c	Только компиляция (без линковки)	gcc -c file.c
-fPIC	Генерация позиционно-независимого кода (для .so)	gcc -fPIC -shared -o lib.so file.c

### 4. Утилита ar

Создает и управляет **статическими библиотеками** (.a).

### 5. Переменная LD\_LIBRARY\_PATH

Список путей для поиска **динамических библиотек** (.so) во время выполнения.

Создание статической библиотеки

```
@ccurecc →.../os_lab_2019/lab2/src/revert_string (master) $ gcc -c revert_string.c -o revert_string.o
@ccurecc →.../os_lab_2019/lab2/src/revert_string (master) $ ar rcs librevert_string.a revert_string.o
@ccurecc →.../os_lab_2019/lab2/src/revert_string (master) $ gcc main.c -I. -L. -ltrevert_string -o static_program
@ccurecc →.../os_lab_2019/lab2/src/revert_string (master) $
```

Запуск файла со статической библиотекой

```
@ccurecc →.../os_lab_2019/lab2/src/revert_string (master) $ ./static_program "Hello World"
Reverted: dlrow olleH
@ccurecc →.../os_lab_2019/lab2/src/revert_string (master) $
```

Создание динамической библиотеки

```
@ccurecc →.../os_lab_2019/lab2/src/revert_string (master) $ gcc -fPIC -c revert_string.c -o revert_string_pic.o
@ccurecc →.../os_lab_2019/lab2/src/revert_string (master) $ gcc -shared -o librevert_string.so revert_string_pic.o
@ccurecc →.../os_lab_2019/lab2/src/revert_string (master) $ gcc main.c -I. -L. -ltrevert_string -o dynamic_program
@ccurecc →.../os_lab_2019/lab2/src/revert_string (master) $
```

Запуск файла с динамической библиотекой

```
@ccurecc →.../os_lab_2019/lab2/src/revert_string (master) $ ./dynamic_program "Hello World"
./dynamic_program: error while loading shared libraries: librevert_string.so: cannot open shared object file: No such file or directory
@ccurecc →.../os_lab_2019/lab2/src/revert_string (master) $ export LD_LIBRARY_PATH=.:$LD_LIBRARY_PATH
@ccurecc →.../os_lab_2019/lab2/src/revert_string (master) $ ./dynamic_program "Hello World"
Reverted: dlrow olleH
@ccurecc →.../os_lab_2019/lab2/src/revert_string (master) $
```

### 1. Статическая библиотека:

ar rcs создает архив (.a) из объектных файлов

При линковке код библиотеки включается в исполняемый файл

Плюсы: не требует наличия библиотеки при запуске

Минусы: увеличивает размер программы

### 2. Динамическая библиотека:

-fPIC генерирует позиционно-независимый код

-shared создает разделяемую библиотеку (.so)

При запуске требуется наличие библиотеки

Плюсы: экономия памяти, возможность обновления без перекомпиляции

### 3. Ключевые опции gcc:

-I. - искать заголовочные файлы в текущей директории

-L. - искать библиотеки в текущей директории

-lrevert\_string - линковаться с librevert\_string.a/so

-o - указать имя выходного файла

### 4. LD\_LIBRARY\_PATH:

Указывает дополнительные пути для поиска динамических библиотек

## Задание 4

### Необходимые знания

3. Переменная окружения `LD_LIBRARY_PATH`

### Задание

Если вы зайдете в `update.sh`, то увидите, следующую строку `sudo apt -y install libcunit1 libcunit1-doc libcunit1-dev`. Это установка в вашу систему библиотеки для юнит тестирования на языке Си - CUnit. В папке `test` вы найдете `tests.c` - готовую программу, которая запускает несколько юнит тестов для функции `RevertString`. Ваша задача, скомпилировать эту программу с динамической библиотекой из **Задания 3**. Важно, что программа с тестами и программа из **Задания 2** должны использовать одну библиотеку (один файл).

P.S. Разумеется, тесты должны проходить :)

```
CUnit - A unit testing framework for C - Version 2.1-3
http://cunit.sourceforge.net/
```

```
Suite: Suite
Test: test of RevertString function ...passed
```

Run Summary:	Type	Total	Ran	Passed	Failed	Inactive
	suites	1	1	n/a	0	0
	tests	1	1	1	0	0
	asserts	4	4	4	0	n/a

```
Elapsed time = 0.000 seconds
```

Тесты не компилировались в папке tests, поэтому я скопировал их в папку из второго и третьего задания

```
@ccurecc →.../os_lab_2019/lab2/src/tests (master) $ gcc tests.c -I. -L. -lrevert_string -lcunit -o test_program
tests.c:5:10: fatal error: revert_string.h: No such file or directory
   5 | #include "revert_string.h"
     | ^~~~~~
compilation terminated.
@ccurecc →.../os_lab_2019/lab2/src/tests (master) $ cd ../
@ccurecc →/workspaces/os_lab_2019/lab2/src (master) $ cp revert_string/revert_string.h tests/
@ccurecc →/workspaces/os_lab_2019/lab2/src (master) $ gcc tests.c -I. -L. -lrevert_string -lcunit -o test_program
gcc: error: tests.c: No such file or directory
@ccurecc →/workspaces/os_lab_2019/lab2/src (master) $ gcc tests/tests.c -I. -L. -lrevert_string -lcunit -o test_program
/usr/bin/ld: cannot find -lrevert_string
collect2: error: ld returned 1 exit status
@ccurecc →/workspaces/os_lab_2019/lab2/src (master) $ cp tests/tests.c revert_string/
@ccurecc →/workspaces/os_lab_2019/lab2/src (master) $ cd revert_string
@ccurecc →.../os_lab_2019/lab2/src/revert_string (master) $ gcc tests.c -I. -L. -lrevert_string -lcunit -o test_program
@ccurecc →.../os_lab_2019/lab2/src/revert_string (master) $ export LD_LIBRARY_PATH=.:$LD_LIBRARY_PATH
@ccurecc →.../os_lab_2019/lab2/src/revert_string (master) $ ./test_program

CUnit - A unit testing framework for C - Version 2.1-3
http://cunit.sourceforge.net/

Suite: Suite
Test: test of RevertString function ...passed

Run Summary:
  Type  Total  Ran  Passed  Failed  Inactive
  suites      1      1      n/a      0      0
  tests       1      1      1      0      0
  asserts     4      4      4      0      n/a

Elapsed time = 0.000 seconds
@ccurecc →.../os_lab_2019/lab2/src/revert_string (master) $
```