0x09. C - Static libraries

C

By Julien Barbier

Weight: 1

Ongoing project - started Jul 22, 2022, must end by Jul 23, 2022 - you're done with 0% of tasks.

✓ Checker was released at Jul 22, 2022 9:00 AM

An auto review will be launched at the deadline

Concepts

For this project, we expect you to look at this concept:

- C static libraries (/concepts/61)

Resources

Read or watch:

- What Is A “C” Library? What Is It Good For? (/rltoken/XB1iH0qE6gshx0x8TfRAPQ)
- Creating A Static “C” Library Using “ar” and “ranlib” (/rltoken/XB1iH0qE6gshx0x8TfRAPQ)
- Using A “C” Library In A Program (/rltoken/XB1iH0qE6gshx0x8TfRAPQ)
- What is difference between Dynamic and Static library(Static and Dynamic linking) (/rltoken/PexOGO-npR_ZDQk-SpOR9g) (*stop at 4:44*)

man or help:

- `ar`
- `ranlib`
- `nm`

Learning Objectives

At the end of this project, you are expected to be able to explain to anyone (/rltoken/dkyFVPYqNQb2pkuknMb9Sw), **without the help of Google**:

General

- What is a static library, how does it work, how to create one, and how to use it

- Basic usage of `ar` , `ranlib` , `nm`

Copyright - Plagiarism

- You are tasked to come up with solutions for the tasks below yourself to meet with the above learning objectives.
- You will not be able to meet the objectives of this or any following project by copying and pasting someone else's work.
- You are not allowed to publish any content of this project.
- Any form of plagiarism is strictly forbidden and will result in removal from the program.

Requirements

C

- Allowed editors: `vi` , `vim` , `emacs`
- All your files will be compiled on Ubuntu 20.04 LTS using `gcc` , using the options `-Wall -Werror -Wextra -pedantic -std=gnu89`
- All your files should end with a new line
- A `README.md` file, at the root of the folder of the project is mandatory
- Your code should use the `Betty` style. It will be checked using betty-style.pl (https://github.com/holbertonschool/Betty/blob/master/betty-style.pl) and betty-doc.pl (https://github.com/holbertonschool/Betty/blob/master/betty-doc.pl)
- You are not allowed to use global variables
- No more than 5 functions per file
- You are not allowed to use the standard library. Any use of functions like `printf` , `puts` , etc… is forbidden
- You are allowed to use _putchar (https://github.com/holbertonschool/_putchar.c/blob/master/_putchar.c)
- You don't have to push `_putchar.c` , we will use our file. If you do it won't be taken into account
- In the following examples, the `main.c` files are shown as examples. You can use them to test your functions, but you don't have to push them to your repo (if you do we won't take them into account). We will use our own `main.c` files at compilation. Our `main.c` files might be different from the one shown in the examples
- The prototypes of all your functions and the prototype of the function `_putchar` should be included in your header file called `main.h`
- Don't forget to push your header file

Bash

- Allowed editors: `vi` , `vim` , `emacs`
- All your scripts will be tested on Ubuntu 20.04 LTS
- All your files should end with a new line (why? (http://unix.stackexchange.com/questions/18743/whats-the-point-in-adding-a-new-line-to-the-end-of-a-file/18789))
- The first line of all your files should be exactly `#!/bin/bash`
- A `README.md` file, at the root of the folder of the project, describing what each script is doing
- All your files must be executable

More Info

You do not need to learn about dynamic libraries, yet.

Quiz questions

Great! You've completed the quiz successfully! Keep going! (Show quiz)

Tasks

0. A library is not a luxury but one of the necessities of life

mandatory

Create the static library `libmy.a` containing all the functions listed below:

```
int _putchar(char c);
int _islower(int c);
int _isalpha(int c);
int _abs(int n);
int _isupper(int c);
int _isdigit(int c);
int _strlen(char *s);
void _puts(char *s);
char *_strcpy(char *dest, char *src);
int _atoi(char *s);
char *_strcat(char *dest, char *src);
char *_strncat(char *dest, char *src, int n);
char *_strncpy(char *dest, char *src, int n);
int _strcmp(char *s1, char *s2);
char *_memset(char *s, char b, unsigned int n);
char *_memcpy(char *dest, char *src, unsigned int n);
char *_strchr(char *s, char c);
unsigned int _strspn(char *s, char *accept);
char *_strpbrk(char *s, char *accept);
char *_strstr(char *haystack, char *needle);
```

If you haven't coded all of the above functions create empty ones with the right prototype.
Don't forget to push your `main.h` file to your repository. It should at least contain all the prototypes of the above functions.

```
julien@ubuntu:~/0x09. Static Librairies$ ar -t libmy.a
0-isupper.o
0-memset.o
0-strcat.o
1-isdigit.o
1-memcpy.o
1-strncat.o
100-atoi.o
2-strchr.o
2-strlen.o
2-strncpy.o
3-islower.o
3-puts.o
3-strcmp.o
3-strspn.o
4-isalpha.o
4-strpbrk.o
5-strstr.o
6-abs.o
9-strcpy.o
_putchar.o
julien@ubuntu:~/0x09. Static Librairies$ nm libmy.a

0-isupper.o:
0000000000000000 T _isupper

0-memset.o:
0000000000000000 T _memset

0-strcat.o:
0000000000000000 T _strcat

1-isdigit.o:
0000000000000000 T _isdigit

1-memcpy.o:
0000000000000000 T _memcpy

1-strncat.o:
0000000000000000 T _strncat

100-atoi.o:
0000000000000000 T _atoi

2-strchr.o:
0000000000000000 T _strchr

2-strlen.o:
0000000000000000 T _strlen

2-strncpy.o:
0000000000000000 T _strncpy

3-islower.o:
0000000000000000 T _islower
```

```
3-puts.o:
                 U _putchar
0000000000000000 T _puts

3-strcmp.o:
0000000000000000 T _strcmp

3-strspn.o:
0000000000000000 T _strspn

4-isalpha.o:
0000000000000000 T _isalpha

4-strpbrk.o:
0000000000000000 T _strpbrk

5-strstr.o:
0000000000000000 T _strstr

6-abs.o:
0000000000000000 T _abs

9-strcpy.o:
0000000000000000 T _strcpy

_putchar.o:
0000000000000000 T _putchar
                 U write
julien@ubuntu:~/0x09. Static Librairies$ cat main.c
#include "main.h"

int main(void)
{
    _puts("\"At the end of the day, my goal was to be the best hacker\"\n\t- Kevin Mitni
ck");
    return (0);
}
julien@ubuntu:~/0x09. Static Librairies$ gcc -std=gnu89 main.c -L. -lmy -o quote
julien@ubuntu:~/0x09. Static Librairies$ ./quote
"At the end of the day, my goal was to be the best hacker"
    - Kevin Mitnick
julien@ubuntu:~/0x09. Static Librairies$
```

Repo:

- GitHub repository: `alx-low_level_programming`
- Directory: `0x09-static_libraries`
- File: `libmy.a, main.h`

☐ Done? Help Check your code

1. Without libraries what have we? We have no past and no future

mandatory

Create a script called `create_static_lib.sh` that creates a static library called `liball.a` from all the `.c` files that are in the current directory.

```
julien@ubuntu:~/0x09. Static Librairies$ ls *.c
0-isupper.c  0-strcat.c  1-isdigit.c  1-strncat.c  2-strlen.c   3-islower.c  3-strcmp.c
4-isalpha.c  5-strstr.c  9-strcpy.c  _putchar.c
0-memset.c   100-atoi.c  1-memcpy.c   2-strchr.c   2-strncpy.c  3-puts.c     3-strspn.c
4-strpbrk.c  6-abs.c
julien@ubuntu:~/0x09. Static Librairies$ ./create_static_lib.sh
julien@ubuntu:~/0x09. Static Librairies$ ls *.a
liball.a
julien@ubuntu:~/0x09. Static Librairies$ ar -t liball.a
0-isupper.o
0-memset.o
0-strcat.o
100-atoi.o
1-isdigit.o
1-memcpy.o
1-strncat.o
2-strchr.o
2-strlen.o
2-strncpy.o
3-islower.o
3-puts.o
3-strcmp.o
3-strspn.o
4-isalpha.o
4-strpbrk.o
5-strstr.o
6-abs.o
9-strcpy.o
_putchar.o
julien@ubuntu:~/0x09. Static Librairies$
```

Repo:

- GitHub repository: `alx-low_level_programming`
- Directory: `0x09-static_libraries`
- File: `create_static_lib.sh`

☐ Done? Help Check your code >_ Get a sandbox