

Langage C - 0x00

Ma librairie statique

Concepts

Pour ce projet, vous êtes appelés à visiter les concepts suivants :

1. man gcc
2. man ar
3. man ranlib
4. Google : C'est quoi une librairie statique

Note : **man** est une commande disponible sur les systèmes d'exploitation de type Unix. Elle permet de visionner les contenus d'une documentation formatée pour être exploitable par man ; à l'origine, elle sert à accéder aux manuels des commandes d'un shell Unix et à la description des fonctions du langage C.

Ressources

Lisez ou regardez :

1. [Tout ce qu'il y a à savoir sur la librairie statique](#)
2. [La différence entre la librairie statique et la librairie dynamique](#)

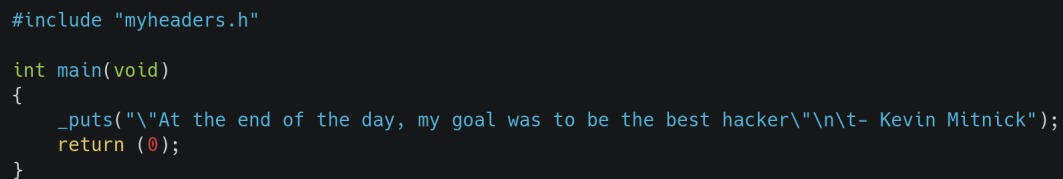
Objectifs d'apprentissage

À la fin de ce projet, vous devez être en mesure d'expliquer à n'importe qui, sans l'aide de Google :

1. Ce qu'est une librairie statique, comment elle fonctionne, comment en créer et comment l'utiliser...
2. Les utilisations basiques de **ar**, **ranlib**, et **nm**

Exigences

- Vous devez travailler sous un système d'exploitation Linux ;
- Vos fichiers seront compilés en utilisant les options **-Wall -Werror -Wextra -pedantic -std=gnu89** ;
- Tous vos fichiers doivent se terminer par une nouvelle ligne vide ;
- Votre code doit suivre le style de coding [Betty](#) ;
 - Cliquez [ici](#) pour voir comment l'installer et tester sur vos fichiers ;
- Vous n'êtes pas autorisé à utiliser des variables globales ;
- Un fichier ne doit pas contenir plus de cinq fonctions ;
- Vous n'avez pas le droit d'utiliser la librairie standard, l'utilisation des fonctions comme printf ou puts est interdite ;
- Vous devez avoir un fichier **myheaders.h** qui contient les prototypes de toutes vos fonctions ;
- Utilisez le code suivant pour tester votre programme :



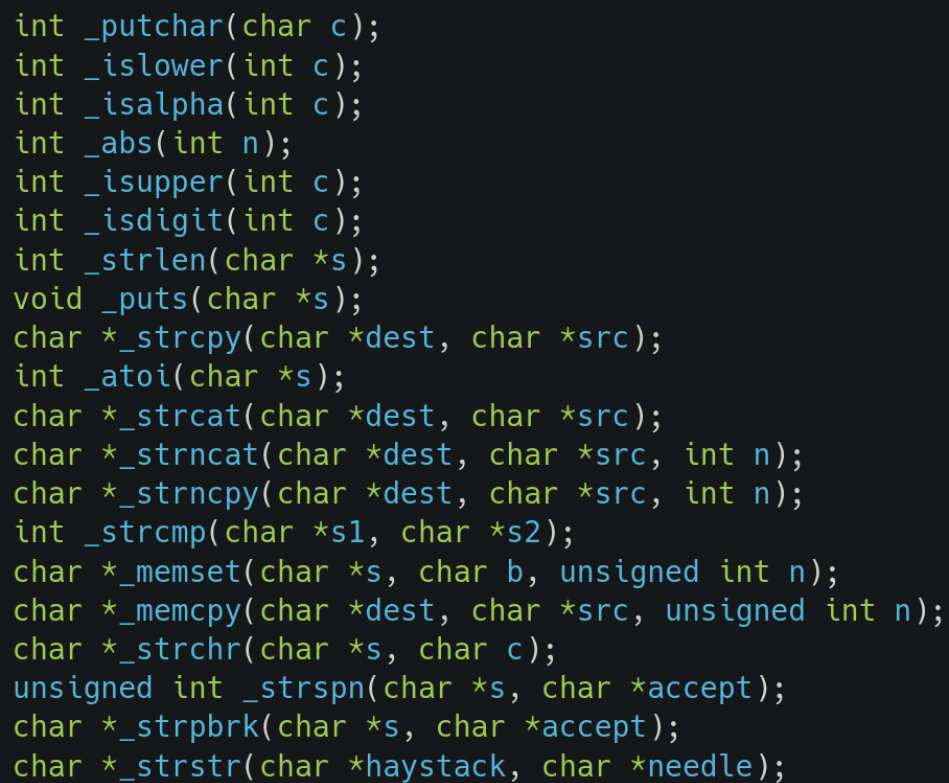
```
#include "myheaders.h"

int main(void)
{
    _puts("\nAt the end of the day, my goal was to be the best hacker\n\t- Kevin Mitnick");
    return (0);
}
```

- Vous pouvez modifier le main pour tester d'autres fonctions ;


Tâches

1. Écrivez le contenu de toutes les fonctions suivantes :



```
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);
```

Ces fonctions existent naturellement en Langage C mais sans le `_` au début. Par exemple, la fonction **islower()** permet de vérifier si un caractère qu'on lui passe en argument est minuscule ou pas. Ainsi, la fonction **_islower()** que vous allez écrire doit faire la même chose. Cherchez la définition des autres fonctions et recodez-les par vos propres moyens sans utiliser les fonctions natives du langage. N'oubliez pas : pas plus de cinq (05) fonctions par fichier.

- 
2. Créez la librairie statique **mylib.a** contenant toutes les fonctions ci-dessus.
 3. Testez votre librairie :
 - a. Exécutez les code suivant sur votre librairie **mylib.a** ; quel est votre output ?
 - i. **ar -t libmy.a**
 - ii. **nm libmy.a**
 4. Créez un fichier **bash** appelé **create_static_lib.sh** qui permet de créer la librairie **mylib.a** une fois exécuté. La première ligne de votre code **Bash** doit être : **#!/bin/bash** . Ce fichier doit contenir exactement trois (03) lignes de code. Il doit être exécutable (man chmod).

Livrables

Vous devez rendre un fichier zip avec un nom sous le format suivant :

NOM_PRENOM_0x00.zip

Bonne chance

