0x09. C - Static libraries

0-putchar.c

#include <unistd.h>

/\*\*

 \* \_putchar - writes the character c to stdout

 \* @c: The character to print

 \*

 \* Return: On success 1.

 \* On error, -1 is returned, and errno is set appropriately.

 \*/

int \_putchar(char c)

{

return (write(1, &c, 1));

}

============================

0-isupper.c

#include "main.h"

/\*\*

 \* \_isupper - uppercase letters

 \* @c: char to check

 \*

 \* Return: 0 or 1

 \*/

int \_isupper(int c)

{

if (c >= 'A' && c <= 'Z')

return (1);

else

return (0);

}

============================

0-memset.c

#include "main.h"

/\*\*

 \* \_memset - fill a block of memory with a specific value

 \* @s: starting address of memory to be filled

 \* @b: the desired value

 \* @n: number of bytes to be changed

 \*

 \* Return: changed array with new value for n bytes

 \*/

char \*\_memset(char \*s, char b, unsigned int n)

{

int i = 0;

for (; n > 0; i++)

{

s[i] = b;

n--;

}

return (s);

}

============================

0-strcat.c

#include "main.h"

/\*\*

 \* \_strcat - concatenates two strings

 \* @dest: input value

 \* @src: input value

 \*

 \* Return: void

 \*/

char \*\_strcat(char \*dest, char \*src)

{

int i;

int j;

i = 0;

while (dest[i] != '\0')

{

i++;

}

j = 0;

while (src[j] != '\0')

{

dest[i] = src[j];

i++;

j++;

}

dest[i] = '\0';

return (dest);

}

============================

1-isdigit.c

#include "main.h"

/\*\*

 \* \_isdigit - checks for a digit (0 through 9)

 \* @c: int to be checked

 \* Return: 1 if c is a digit, 0 otherwise

 \*/

int \_isdigit(int c)

{

return (c >= '0' && c <= '9');

}

============================

1-memcpy.c

#include "main.h"

/\*\*

 \*\_memcpy - a function that copies memory area

\*@dest: memory where is stored

\*@src: memory where is copied

\*@n: number of bytes

\*

\*Return: copied memory with n byted changed

\*/

char \*\_memcpy(char \*dest, char \*src, unsigned int n)

{

int r = 0;

int i = n;

for (; r < i; r++)

{

dest[r] = src[r];

n--;

}

return (dest);

}

============================

1-strncat.c

#include "main.h"

/\*\*

 \* \_strncat - concatenate two strings

 \* using at most n bytes from src

 \* @dest: input value

 \* @src: input value

 \* @n: input value

 \*

 \* Return: dest

 \*/

char \*\_strncat(char \*dest, char \*src, int n)

{

int i;

int j;

i = 0;

while (dest[i] != '\0')

{

i++;

}

j = 0;

while (j < n && src[j] != '\0')

{

dest[i] = src[j];

i++;

j++;

}

dest[i] = '\0';

return (dest);

}

============================

2-strchr.c

#include "main.h"

/\*\*

 \* \_strchr - Entry point

 \* @s: input

 \* @c: input

 \* Return: Always 0 (Success)

 \*/

char \*\_strchr(char \*s, char c)

{

int i = 0;

for (; s[i] >= '\0'; i++)

{

if (s[i] == c)

return (&s[i]);

}

return (0);

}

============================

2-strlen.c

#include "main.h"

/\*\*

 \* \_strlen - returns the length of a string

 \* @s: string

 \* Return: length

 \*/

int \_strlen(char \*s)

{

int longi = 0;

while (\*s != '\0')

{

longi++;

s++;

}

return (longi);

}

============================

2-strncpy.c

#include "main.h"

/\*\*

 \* \_strncpy - copy a string

 \* @dest: input value

 \* @src: input value

 \* @n: input value

 \*

 \* Return: dest

 \*/

char \*\_strncpy(char \*dest, char \*src, int n)

{

int j;

j = 0;

while (j < n && src[j] != '\0')

{

dest[j] = src[j];

j++;

}

while (j < n)

{

dest[j] = '\0';

j++;

}

return (dest);

}

============================

3-islower.c

#include "main.h"

/\*\*

 \* \_islower - checks for lowercase character

 \* @c: the character to check

 \* Return: 1 if c is lowercase, 0 otherwise

 \*/

int \_islower(int c)

{

return (c >= 'a' && c <= 'z');

}

== == == == == == == == == == == == == ==

    3 - puts.c

#include "main.h"

#include <stdio.h>

/\*\*

 \* main - Entry point

 \*

 \* Return: Always 0 (Success)

 \*/

void \_puts(char \*str)

{

int i = 0;

while (str[i])

{

\_putchar(str[i]);

i++;

}

\_putchar('\n');

}

============================

3-strcmp.c

#include "main.h"

/\*\*

 \* \_strcmp - compare string values

 \* @s1: input value

 \* @s2: input value

 \*

 \* Return: s1[i] - s2[i]

 \*/

int \_strcmp(char \*s1, char \*s2)

{

int i;

i = 0;

while (s1[i] != '\0' && s2[i] != '\0')

{

if (s1[i] != s2[i])

{

return (s1[i] - s2[i]);

}

i++;

}

return (0);

}

============================

3-strspn.c

#include "main.h"

/\*\*

 \* \_strspn - Entry point

 \* @s: input

 \* @accept: input

 \* Return: Always 0 (Success)

 \*/

unsigned int \_strspn(char \*s, char \*accept)

{

unsigned int n = 0;

int r;

while (\*s)

{

for (r = 0; accept[r]; r++)

{

if (\*s == accept[r])

{

n++;

break;

}

else if (accept[r + 1] == '\0')

return (n);

}

s++;

}

return (n);

}

============================

4-isalpha.c

#include "main.h"

/\*\*

 \* \_isalpha - checks for alphabetic character

 \* @c: the character to be checked

 \* Return: 1 if c is a letter, 0 otherwise

 \*/

int \_isalpha(int c)

{

return ((c >= 'a' && c <= 'z') || (c >= 'A' && c <= 'Z'));

}

============================

4-strpbrk.c

#include "main.h"

/\*\*

 \* \_strpbrk - Entry point

 \* @s: input

 \* @accept: input

 \* Return: Always 0 (Success)

 \*/

char \*\_strpbrk(char \*s, char \*accept)

{

int k;

while (\*s)

{

for (k = 0; accept[k]; k++)

{

if (\*s == accept[k])

return (s);

}

s++;

}

return ('\0');

}

============================

5-strstr.c

#include "main.h"

/\*\*

 \* \_strstr - Entry point

 \* @haystack: input

 \* @needle: input

 \* Return: Always 0 (Success)

 \*/

char \*\_strstr(char \*haystack, char \*needle)

{

for (; \*haystack != '\0'; haystack++)

{

char \*l = haystack;

char \*p = needle;

while (\*l == \*p && \*p != '\0')

{

l++;

p++;

}

if (\*p == '\0')

return (haystack);

}

return (0);

}

============================

6-abs.c

#include "main.h"

/\*\*

 \* \_abs - computes the absolute value of an integer

 \* @n: the int to check

 \* Return: the absolute value of int

 \*/

int \_abs(int n)

{

if (n >= 0)

{

return (n);

}

return (-n);

}

============================

9-strcpy.c

#include "main.h"

/\*\*

 \* char \*\_strcpy - a function that copies the string pointed to by src

 \* @dest: copy to

 \* @src: copy from

 \* Return: string

 \*/

char \*\_strcpy(char \*dest, char \*src)

{

int l = 0;

int x = 0;

while (\*(src + l) != '\0')

{

l++;

}

for ( ; x < l ; x++)

{

dest[x] = src[x];

}

dest[l] = '\0';

return (dest);

}

============================

100-atoi.c

#include "main.h"

/\*\*

 \* \_atoi - convert a string into an integer.

 \*

 \* @s: the string to use.

 \*

 \* Return: integer.

 \*/

int \_atoi(char \*s)

{

int sign = 1, i = 0;

unsigned int res = 0;

while (!(s[i] <= '9' && s[i] >= '0') && s[i] != '\0')

{

if (s[i] == '-')

sign \*= -1;

i++;

}

while (s[i] <= '9' && (s[i] >= '0' && s[i] != '\0'))

{

res = (res \* 10) + (s[i] - '0');

i++;

}

res \*= sign;

return (res);

}

============================

main.h

#ifndef MAIN\_H

#define MAIN\_H

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);

#endif

============================

C - Static libraries

TASKS

[0. A library is not a luxury but one of the necessities of life](libmy.a)

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

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.

[1. Without libraries what have we? We have no past and no future](create\_static\_lib.sh)

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.