C1- Unix & C Lab Seminar

pool_c_d05

Day 05

Point It Further !

CODING ACADEMY
> BY EPITECH

Instructions

Before You Go Further

- Turn in directory **pool_c_d05**.
- Respecting the norm takes time, but it is good for you. This way your code will respect the norm from the first written line to the last.
  Read carefully the norm documentation. You should type "alt+i" instead of "tab"
- Do not care about the header. You don't have to add it into your file.
- You shall leave in your directory no other files than those explicitly specified by the exercices. If one of your files prevents the compilation with *.c, the robot will not be able to do the correction and you will have a 0.
- You might have to use the NULL keyword. This keywork is located in seddef.h
- **Do not turn-in a main() function unless it has been explicitly asked.**
- **You are only allowed to use the write function to do the following exercices.**
- **Remeber that your exercises should never creash, to prevent that you should always test every cases of functionalities (like empty strings for example)**

> Remember it is always better to create your repository at the beginning of the day and to turn-in your work on a regular basis. Do not forget the permissions rights!

Exercise 1

array_sum (2pts)

Turn in: pool_c_d05/ex_01/ex_01.c
Prototype: int array_sum(int *tab, int size);

Write a function 'array_size', taking an array of interger and its size as parameter. This function must return the sum of the integers containd in the array.

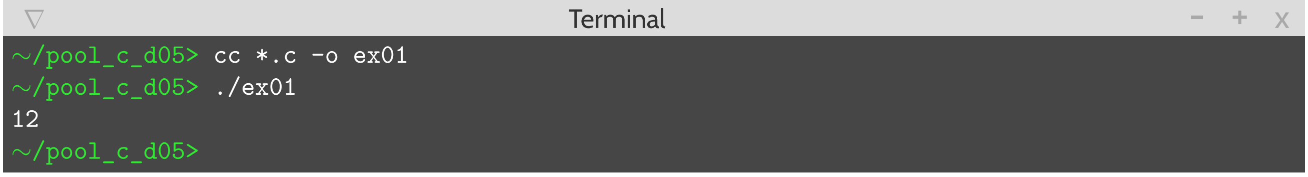
Example :
main.c :

```
#include <stdio.h>

int     array_sum(int *tab, int size);

int     main()
{
  int   tab[3] = {3, 4, 5};

  printf("%d\n", array_sum(tab, 3));
  return (0);
}
```



```
~/pool_c_d05> cc *.c -o ex01
~/pool_c_d05> ./ex01
12
~/pool_c_d05>
```

Exercise 2

my_revstr (3pts)

Turn in: pool_c_d05/ex_02/ex_02.c
Prototype: char *my_revstr(char *str);

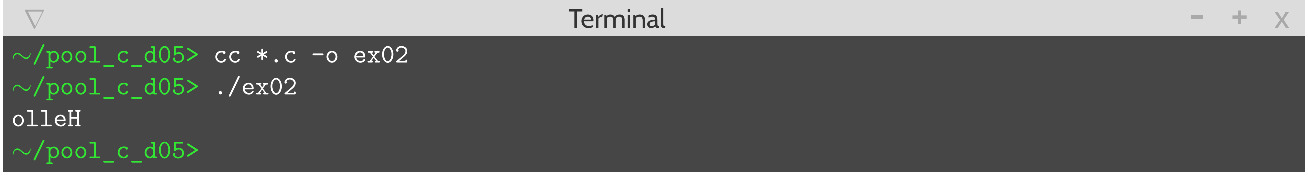
Write a function 'my_revstr', taking a string as parameter and returning it in reverse.

Example :
main.c :

```
#include <stdio.h>

char    *my_revstr(char *str);
int     main()
{
  char  str[6] = "Hello";

  printf("%s\n", my_revstr(str));
}
```



```
~/pool_c_d05> cc *.c -o ex02
~/pool_c_d05> ./ex02
olleH
~/pool_c_d05>
```

Exercise 3

my_strcmp (4pts)

Turn in: pool_c_d05/ex_03/ex_03.c
Prototype: int my_strcmp(char *str1, char *str2);

Write a function 'my_strcmp', taking two string as parameters. Reproducing the behavior of the function "strcmp", Careful, some strcmp function only return (-1, 0 or 1), that is not the behavior that we expect.

man strcmp;

Exercise 4

my_strupcase (2pts)

Turn in: pool_c_d05/ex_04/ex_04.c
Prototype: char *my_strupcase(char *);

Write a function 'my_strupcase', taking a string as parameter and returning the string with every letter in uppercase.

Exercise 5

my_str_isalpha (2pts)

Turn in: pool_c_d05/ex_05/ex_05.c
Prototype: int my_str_isalpha(char *);

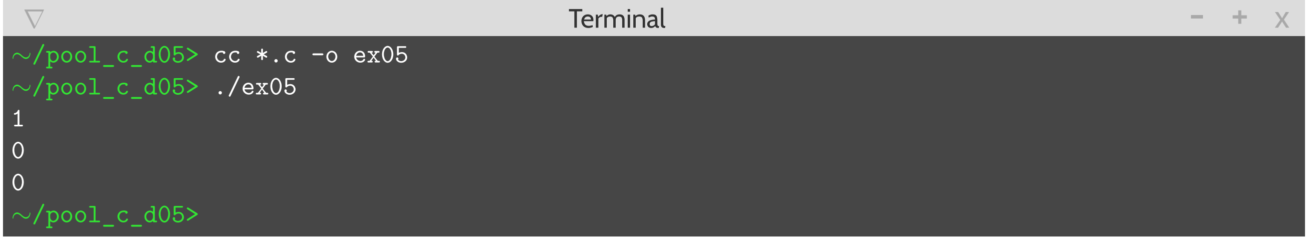
Write a function 'my_str_isalpha', taking a string as parameter and returning 1 if the string passed as parameter contains only alphabetical characters, otherwise it should return 0.

Example :
main.c :

```
#include <stdio.h>

int     my_str_isalpha(char *str);
int     main()
{
  char  str1[6] = "Hello";
  char  str2[7] = "Pony42";
  char  str3[3] = "42";

  printf("%d\n", my_str_isalpha(str1));
  printf("%d\n", my_str_isalpha(str2));
  printf("%d\n", my_str_isalpha(str3));
}
```



```
~/pool_c_d05> cc *.c -o ex05
~/pool_c_d05> ./ex05
1
0
0
~/pool_c_d05>
```

Exercise 6

my_strstr (3pts)

Turn in: pool_c_d05/ex_06/ex_06.c
Prototype: char *my_strstr(char *, char *);

Write a function 'my_strstr' reproducing the behavior of the function strstr.

man strstr;

Exercise 7

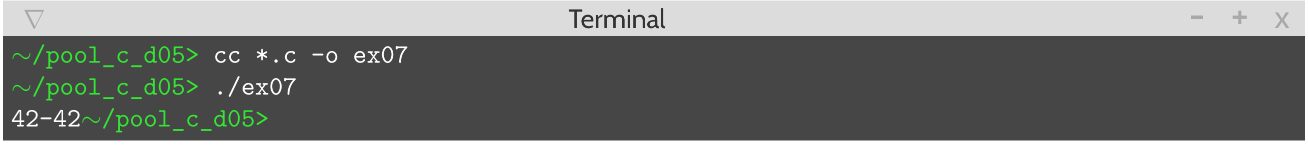
my_putnbr (1pts)

Turn in: pool_c_d05/ex_07/ex_07.c
Prototype: void my_putnbr(int n);

Write a function 'my_putnbr' taking a number as parameter. This function will print this number on the standard output.

Example :
main.c :

```
void    my_putnbr(int n);
int     main()
{
  my_putnbr(42);
  my_putnbr(-42);
  return (0);
}
```



```
~/pool_c_d05> cc *.c -o ex07
~/pool_c_d05> ./ex07
42-42~/pool_c_d05>
```

Exercise 8

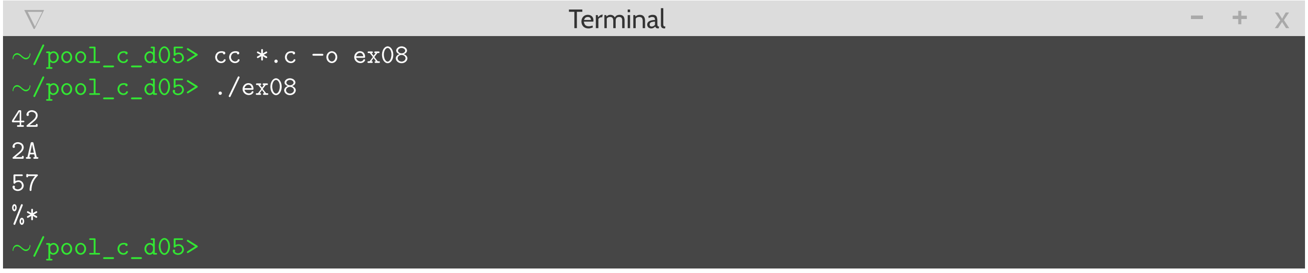
my_putnbr_base (3pts)

Turn in: pool_c_d05/ex_08/ex_08.c
Prototype: int my_putnbr_base(int nbr, char *base);

Write a function 'my_putnbr_base', taking a number as it first parameter and a string as it second parameter representing a base. Your function must display the number in the given base. The function will always return the number passed as parameter.

Example :
main.c :

```
int      my_putnbr_base(int nbr, char *base);
int      main()
{
  my_putnbr_base(42, "0123456789");
  my_putnbr_base(42, "0123456789ABCDEF");
  my_putnbr_base(42, "9876543210");
  my_putnbr_base(42, "+-*/%()-_=");
}
```



```
~/pool_c_d05> cc *.c -o ex08
~/pool_c_d05> ./ex08
42
2A
57
%*
~/pool_c_d05>
```