C1 - Pool C

C-CPE-042

Day 06

Play With Address

1.1

EPITECH.

Day 06

repository name: pool_c_d06
repository rights: ramassage-tek

- The totality of your source files, except all useless files (binary, temp files, obj files,...), must be included in your delivery.

+ Instructions: Before You Go Further

- Turn in directory: **pool_c_d06**.
- 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”
- 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 stddef.h
- **Do not turn-in a main() function unless it has been explicitly asked.**
- **From this day forward, you are allowed to use the function ‘malloc’. You are allowed to use the function ‘printf’ only when it’s explicitly asked. The others functions are forbidden.**
- **Note: Today we will not give you the Prototype of the function we are waiting. You will have to read carefully the subject as to find by yourself the prototype of the function.**
- **In today exercises, you will also need to write “#include <stdlib.h>” at the top of some files.**

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: MY_STRCPY (3PTS)

Turn in: pool_c_d06/ex_01/ex_01.c

Write a function 'my_strcpy' taking two pointers to a char as parameters. The first one is the destination and the second one is the source. Your function must copy the source into the destination including the terminating '\0'. The function must return a pointer to char which is the destination parameter.

The parameter used as destination will always be large enough to copy source into it in our tests, you don't have to handle this case in your code.

+ EXERCISE 2: MY_SHOW_ADDRESS (2PTS)

Turn in: pool_c_d06/ex_02/ex_02.c
Function allowed for this exercise: printf

Write a function 'my_show_address' taking a pointer of number as parameter and returning nothing. By using 'printf', your function must display the memory address of the variable in hexadecimal followed by a newline.

+ EXERCISE 3: MY_UP (4PTS)

Turn in: pool_c_d06/ex_03/ex_03.c

Write a function 'my_up' taking an integer as parameter. This function must return an allocated array of two elements. The first element must be the number given in the parameter and the second one the same number multiplied by 2.

+ EXERCISE 4: MY_STRDUP (2PTS)

Turn in: pool_c_d06/ex_04/ex_04.c

Write a function 'my_strdup', taking a string in parameter. Your function must return a copy of the string in a new allocated memory space.

+ EXERCISE 5: MY_SHOW_STR (2PTS)

Turn in: pool_c_d06/ex_05/ex_05.c

Write a function 'my_show_str', taking a double array of chars as parameter and returning nothing. Your function must display each string of chars contained in the array followed by a newline. The array will be terminated by a null value.

Example :

Terminal

```
~/C-CPE-042> cat main.c
#include <stdlib.h>

void my_show_str(char **);
int main()
{
     char *tab[] = {
          "Hello",
          "Students",
          NULL
     };
     my_show_str(tab);
     return(0);
}
```

Terminal

```
~/C-CPE-042> cc *.c -o ex05
~/C-CPE-042> ./ex05
Hello
Students
~/C-CPE-042>
```

+ EXERCISE 6: MY_CONCAT_STR (3PTS)

Turn in: pool_c_d06/ex_06/ex_06.c

Write a function 'my_concat_str', taking a double array of char as parameter. Your function will return a sufficiently allocated string of char terminated by a textbackslash 0', which contains each word of the array concatenated one after another. The array will be terminated by a null value.

Example :

Terminal

```
~/C-CPE-042> cat main.c
#include <stdlib.h>

void my_putstr(char *);
char *my_concat_str(char **);
int main()
{
     char *tab[] = {
          "Hello",
          "Students\n",
          NULL
     };
     char *str;

     str = my_concat_str(tab);
     my_putstr(str);
     return(0);
}
```

Terminal

```
~/C-CPE-042> cc *.c -o ex06
~/C-CPE-042> ./ex06
HelloStudents
~/C-CPE-042>
```

+ EXERCISE 7 (MY_RESET_PTR (4PTS))

Turn in: pool_c_d06/ex_07/ex_07.c

Write a function 'my_reset_ptr' taking a pointer to a pointer of char as parameter. This function will return nothing and must free the pointer pointed by the parameter. After freeing it you must set this pointer to NULL.

Example :

Terminal

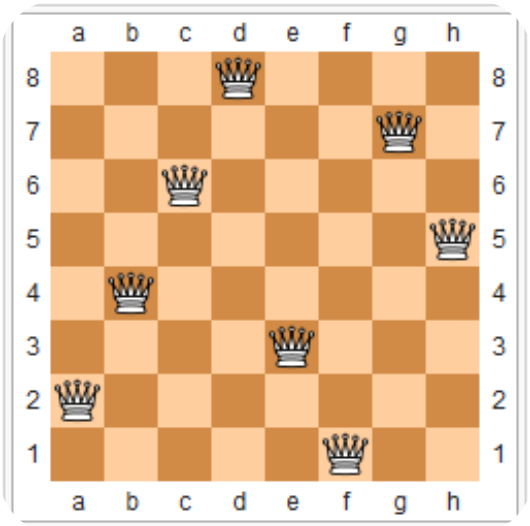
```
~/C-CPE-042> cat main.c
#include <string.h>

void my_reset_ptr(char **ptr);
int main()
{
     char *str;

     str = strdup("Please, let me free!");
     my_reset_ptr(&str);
     return (0);
}
```

+ Exercise Bonus (8 Queens (5pts))

Turn in: pool_c_d06/ex_08/my_8_queens.c
Prototype: int my_8_queens();

Write a function that displays every possibility of placing 8 queens on a chessboard without them being able to tun into each other's in a single move.

The display will be as follow for the following solution example (don't leave your main of test for the turn-in):



Example :

Terminal

```
~/C-CPE-042> cc *.c -o my_8_queens
~/C-CPE-042> ./my_8_queens
d8,g7,c6,h5,b4,e3,a2,f1
[...]
~/C-CPE-042>
```

In this example, we only have one solution to the "8 queens" problem. You need to find and display all possibilitied to succeed in the exercise.

There is a line break after each solution of the 8 queens' problem.