Subject(s):

- Pointers.
- Function call by value and by reference

Part 1

1. Implement the code present on pages 11 and 12 of the theoretical slides. Debug and try to understand how it works.
2. Complete the following code following the indications given by the comments. Analyse the behaviour of the program.

```
#include <stdio.h>

#include <stdio.h>

void foo(int a, int *aPtr) {

    // Print the value of a
    // print the address of a (local)

    // Print the value of aPtr
    // Print the address of aPtr

    // sum 1 to a Print the value of
    // sum 1 to the value pointed by aPtr
}

int main() {
    int a = 0;
    // Print the value of a
    // print the address of a
    foo(a, &a);
    // Print the value of a
    return 0;
}
```

3. Implement a program that reads 2 integer values in the main function and then pass these values to the function **`void sum (int num1, int num2, long *result)`**. This function should return the sum of the 2 values in the **`result`** parameter, which should be presented in the **`main`** function.

4. Implement the function `void sort(int *v1, int *v2, int *v3)` that receives 3 pointers to integer variables and sort the values. For example: `sort(1, 3, 2)` must assign to `v1` the value `1`, `v2` the value `2` and `v3` the value `3`.
5. Implement the `int len(char *str)` function that returns the size to the string without using libraries other than `stdio.h`.

Part 2

1. Implement the function `int equal (char *str1, char *str2)` which returns `1` if the strings are equal or `0` if they are different.
2. Implement the function `void printFirst(char *str, int n)` that prints character by character the first `n` characters of a string.
3. Implement the function `char *strchar(char *str, char ch)` that returns the address of the first occurrence of `ch` in `str`. If it does not exist, return `NULL`.
4. Implement the function `int *findMax(int data[], int size)` that receives an array and its size. The function must return the memory address of the array element that has the highest value.
5. The `swap` function seems to work, but not `swap_pointers`. Fix it.

```
#include <stdio.h>

void swap(int *x, int *y) {
    int tmp = *x;
    *x = *y;
    *y = tmp;
}

void swap_pointers(char *x, char *y) {
 char *tmp;
 tmp = x;
 x = y;
 y = tmp;
}

int main() {
    int a = 0, b = 1;
    char *s1 = " fundamentals", *s2 = "Programming";

    swap(&a, &b);
    printf("a: %d\n", a);
    printf("b: %d\n", b);

    swap_pointers(s1, s2);
    printf("%s %s\n", s1, s2);

    return 0;
}
```