THREAD

Verificare se la stringa in input è palindroma

```
#include <pthread.h>   #include <string.h>
#include <stdio.h>
#include <unistd.h>

struct print_params {
    char word[];
    int position;
};

int main() {
    pthread_t thread1_id;
    pthread_t thread2_id;

    char* thread1_returnValue;
    char* thread2_returnValue;

    struct print_params thread1_args;
    struct print_params thread2_args;

    char string[] = "osso";
    int centro = (int) (strlen(string)/2);

    thread1_args.word = string;
    thread2_args.word = string;
    thread1_args.position = centro;
    thread2_args.position = centro + 1;

    pthread_create(&thread1_id, NULL, &print_string_1, &thread1_args);
    pthread_create(&thread2_id, NULL, &print_string_2, &thread2_args);
```

due thread perché uno verifica la prima metà della stringa e l'altro la seconda metà

VALORE DEI THREAD — 0 = ritorno

Passo ai thread gli argomenti

VEDI FOGLIO

VOID FOCUS

```
pthread_join (thread1_id, (void**) &thread1_returnValue);
pthread_join (thread2_id, (void**) &thread2_returnValue);

if (thread1_returnValue == thread2_returnValue) {
        printf("La stringa è palindroma");
} else {
        printf("La stringa non è palindroma");
}
}

void* print_string_1 (void* parameters) {

        struct print_params* pp = (struct print_params*) parameters;
        char buffer[pp->position];                          → CENTRO
        for (int i=0; i<pp->position; i++) {
            buffer[i] = pp->word[i];
        }                                  Buffer → Stringa che ritornano di Word
        return (void*) &buffer;
}

void* print_string_2 (void* parameters) {

int c=0                 struct print_params* pp = (struct print_params*) parameters;
                        char buffer[(sizeof(pp->word) - pp->position) + 1];   → per andare da 4 a 7
                        for (int i = sizeof(pp->word) ; i > position; i--) {   Senso di stampa
                            buffer[c] = pp->word[i]; c++;
                        }
                        return (void*) &buffer;
}
```