C programming excercices 1

1 Beginner excercises

1. Create a hello world example.
2. Create a hello world example with defineing two macros. One beeing `HELLO`, the other `WORLD`
3. Lest check the default value of a variable. Create a variable and print out its value, without giving it one.
4. Print out the address and the value of the previous variable in the following format:

```
[address]:    value
[0xABABABAB]: 00123
```

5. Print out the address and the value of a floating-point variable in the following format:

```
[0xABABABAB]: 012.345
```

6. Create a variable containing the following string: `"The sky is beautiful!"`. Print it out until the 10th character.
7. Replace the `'e'` characters in the previous string to the `'X'` character.
8. Create a struct with the following parameters `"int index; int value;"`. Define it as a type. Create an array of the previously defined type with 10 elements, and fill it up with the square of 1 to 10. Print out the values of the array after filling it up.
9. Create the function `void square(int *a)` and square the incoming pointers value. Print out the value of A before and after executing said function.
10. Create character array, with 100 elements. Ask the user for a text input, and read it into the created array. Count the characters given by the user and copy the content into a new char "array", which has the exact size of the input text. *Make sure to also free the new array before exiting the program.*

2 Harder excercises

1. Create a function `void replacer(char* str)`, which will write `"FILTERED"` over the second word of the text. Make sure to don't overflow the original variable. The text can cover multiple words, until the end of the incoming string, but only need to be written once. Print out the incoming string before and after the function execution.

2. Iterate trough the following array until we are at the last elemnt, containing a `NULL (0)` value:

```
unsigned int myarray[] = { 1, 2, 3, 4, 5, 11, 22, 33, 44, 55, 0 };
```

   and print out its values.

3. Create a simple Linked List. Complete the following code and test your approach.

```
struct list {
    int item;
    struct list* next;
};

struct list* list_create() {
    /* TODO */
}
int list_add(struct list*, int item) {
    /* TODO */
}
int list_remove(struct list*, int index) {
    /* TODO */
}
int list_get(struct list*, int index) {
    /* TODO */
}
struct list* list_get_next(struct list*) {
    /* TODO */
}

int main() {
    struct list* my_list = list_create();
    /* TODO */
}
```