Assignment 8

Exercise 00: vc_boolean.h

Turn-in files	vc_boolean.c
Allowed functions	nothing

- Create a vc_boolean.h file. It will compile and run the following main appropriately.

```
#include <stdio.h>
#include "vc_boolean.h"

void vc_putstr(char *str)
{
    while (*str)
        write(1, str++, 1);
}

t_bool vc_is_even(int n)
{
    return ((EVEN(n)) ? TRUE : FALSE);
}

int main(int argc, char **argv)
{
    (void)argv;
    if (vc_is_even(argc - 1) == TRUE)
        vc_putstr(EVEN_MSG);
    else
        vc_putstr(ODD_MSG);
    return SUCCESS;
}
```

- This program should display

```
I have an even number of arguments.
```

- or

```
I have an odd number of arguments.
```

- Followed by a line break.

Exercise 01: vc_abs.h

Turn-in files	vc_abs.h
Allowed functions	nothing

- Create a macro **ABS** which replaces its argument by its absolute value:

```
#define ABS(value)
```

Exercise 02: vc_point.h

Turn-in files	vc_point.h
Allowed functions	nothing

- Create a file **vc_point.h** that will compile the following main

```
#include "vc_point.h"

void set_point(t_point *point)
{
    point->x = 86;
    point->y = 89;
}

int main(void)
{
    t_point point;

    set_point(&point);
    return 0;
}
```

Exercise 03: vc_param_to_tab

Turn-in files	vc_param_to_tab.c, vc_stock.h
Allowed functions	vc_split_whitespaces, vc_strdup, malloc

- Create a function that stores the program's arguments **av** within an array of structs and that returns the address of that array's first box.
- All elements of the array must be processed, including **av[0]**.
- Here's how it should be prototyped:

```
struct s_stock *vc_param_to_tab(int ac, char **av);
```

- The struct array should be allocated and its last box shall contain 0 in its **str** element to point out the end of the array.
- The struct is defined in the **vc_stock.h** file, like this:

```
typedef struct s_stock
{
    int size_param;
    char *str;
    char *copy;
    char **words;
} t_stock;
```

- *size_param* is the length of the argument;
- *str* is the address of the argument;
- *copy* is the copy of the argument;
- *words* is the array returned by **vc_split_whitespaces**.

Exercise 04: vc_show_tab

Turn-in files	**vc_show_tab.c, vc_stock.h**
Allowed functions	putchar

- Create a function that displays the content of the array created by the previous exercise.
- Here's how it should be prototyped:

```
void vc_show_tab(struct s_stock *stock);
```

- For each box, you'll have to display (one per line):
  - the argument
  - the size
  - each word (one per line)

Exercise 05: hexdump

Turn-in files	**vc_hexdump.c**
Allowed functions	close, open, read, write, malloc, free

- Create a program called **vc_hexdump** which does the same thing as the system's **hexdump** command-line.
- The only option you have to handle is **-C**.
- You may use the variable **errno**.