From pointer to GDB my H.3.R.0

1 Exercises

1.1 Ex 1

Using GDB, try to fix this function.

```c
#include <stdio.h>

void swap(int *xp, int *yp)
{
    int temp = *xp;
    *xp = *yp;
    *yp = temp;
}

void bubbleSort(int arr[], int n)
{
    for (int i = 0; i < n - 1; i++)

    for (int j = 0; j < n - i - 1; j++)
        if (arr[j] > arr[j + 1])
            swap(&arr[j], &arr[j + 1]);
}
//Do not use a print function. Use only GDB commands.
int main(void)
{
    int arr[] = {64, 34, 25, 12, 90, 11, 63};
    bubbleSort(arr, 6);
    return 0;
}
```

1.2 Ex 2

Using Valgrind, try to fix this function.

```c
#include <stdlib.h>
void print_array(int *tab, int size)
{
    for (int i = 0; i <= size; i++)
        printf("%d\n", tab[i]);
}

int main()
{
    int *arr = malloc(5 * sizeof(int));
    print_array(arr, 5);
}
```

1.3 Linked List

```c
typedef struct s_list
{
    struct s_list *next;
    void *data;
} t_list;
```

Using this structure, create the functions bellow

1.3.1 Create list

```
t_list *create_elem(void *data);
```

This function create a new element t_list.

1.3.2 Push element at the end

```
void list_push_end(t_list **begin_list, void *data);
```

This function add a new element to the end of the t_list.

1.3.3 Push element at the beginning

```
void list_push_front(t_list **begin_list, void *data);
```

This function add a new element to the beginning of the t_list.

1.3.4 Push at index

```
void list_push_idx(t_list **begin_list, void *data, int indx);
```

This function add a new element in an index.
if index is greater than the size of the list, push at the end.

1.3.5 Clear at index

```
void list_delete_element(t_list **begin_list, int indx);
```

This function delete an element at the index.
if index is greater than the size of the list, do nothing.

1.4 C library

1.4.1 What is

```
int rk_isalpha(int c);
int rk_isdigit(int c);
int rk_isalnum(int c);
int rk_isspace(int c);
```

man 3 isalpha/isdigit/isalnum/isspace.

1.4.2 Memset

```
void *rk_memset(void *s, int c, size_t n);
```

Man 3 memset.

1.4.3 Memcpy

```
void *rk_memcpy(void *dest, const void *src, size_t n);
```

Man 3 memcpy.

1.4.4 Strdup

```
char *rk_strdup(const char *s);
char *rk_strndup(const char *s, size_t n);
```

Man 3 strdup.