SAE S1.02

The objective of this Sae, is to sort in alphabetical order a file containing a list of students, with two different methods

```c
#define TAILLE_MAX 20
#define N 500
```

Data Structures

```c
typedef struct{
  1 char prenom[TAILLE_MAX];
  2 char nom[TAILLE_MAX];
  3 int id;
}etudiant;
```

File_in

1 prenom 2 nom 3 id

```
Oscar    FERLAIN 37700
```

Sorting Algorithm

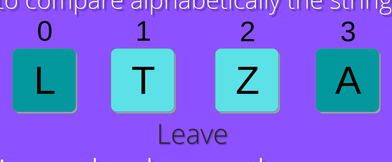
Bubble Sort

A Bubble Sort involves of the following steps

Compare the first two elements in the array.

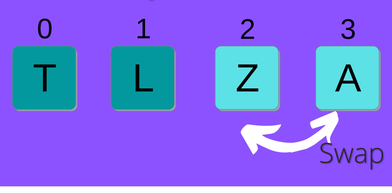
If their are not in order, Swap them.

We use strcmp to compare alphabetically the strings



If they are in order leave them as they are .

Repeat the first step for the rest of the element in the array.



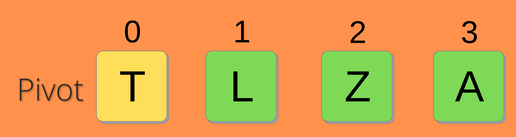
After going through the whole array once, repeat the process until no exchange is needed, this means that the array is sorted

A bubble sort can be realy slow if the array is large and very disordered

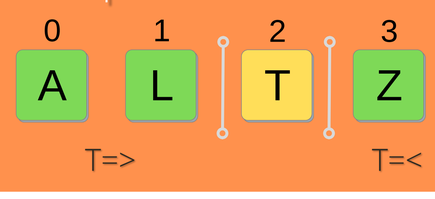
Quick Sort

A Quick Sort is based on the divide and conquer approach it involves of the following steps

An list is divided into subarrays by selecting a pivot element (In this example is T).



While dividing the array, the pivot element should be positioned in such a way that elements less than pivot are kept on the left side and elements greater than pivot are on the right side of the pivot.



The left and right subarrays are also divided using the same approach. This process continues until each sub-lists contains a single element.

this means that the list is sorted

In this example the quick sort already sort the array

A quick sort as its name indicates is fast and does not use much memory

```c
function passageBulle(etudiant: T, int: DEB, FIN)
    int: compt
start
    compt <- 0
    for i from DEB to FIN
        if T[i].nom > T[i+1].nom
            echange(T[i], T[i+1])
            compt <- compt + 1
        endif
    endfor
    return compt
end

procédure triBulle(etudiant: T)
    int: echange, DEB, FIN
start
    DEB <- 0
    FIN <- N-2
    do
        echange <- passageBulle(T, DEB, FIN)
    while echange != 0
end
```

```c
function partition(etudiant T, int : DEB, FIN):int
    int : i, compt
    char : pivot[TAILLE_MAX]
    pivot = T[DEB].nom
    compt = DEB

    Début
    for i from DEB + 1 to FIN
        if T[i].nom < pivot
            compt = compt + 1
            echange(T[compt], T[DEB])
        endif
    endfor
    echange(T[compt], T[DEB])
    return compt

procedure Tri_rapide(etudiant T[], int : DEB, FIN)
    int : p
    if (DEB<FIN)
        p = partition(T,DEB,FIN)
        Tri_rapide(T,DEB,p - 1)
        Tri_rapide(T,p+1,FIN)
    endif
```

Comparison with timer

```
Execution time : 0.0150000
```

```
Execution time : 0.0040000
```