Technical Documentation

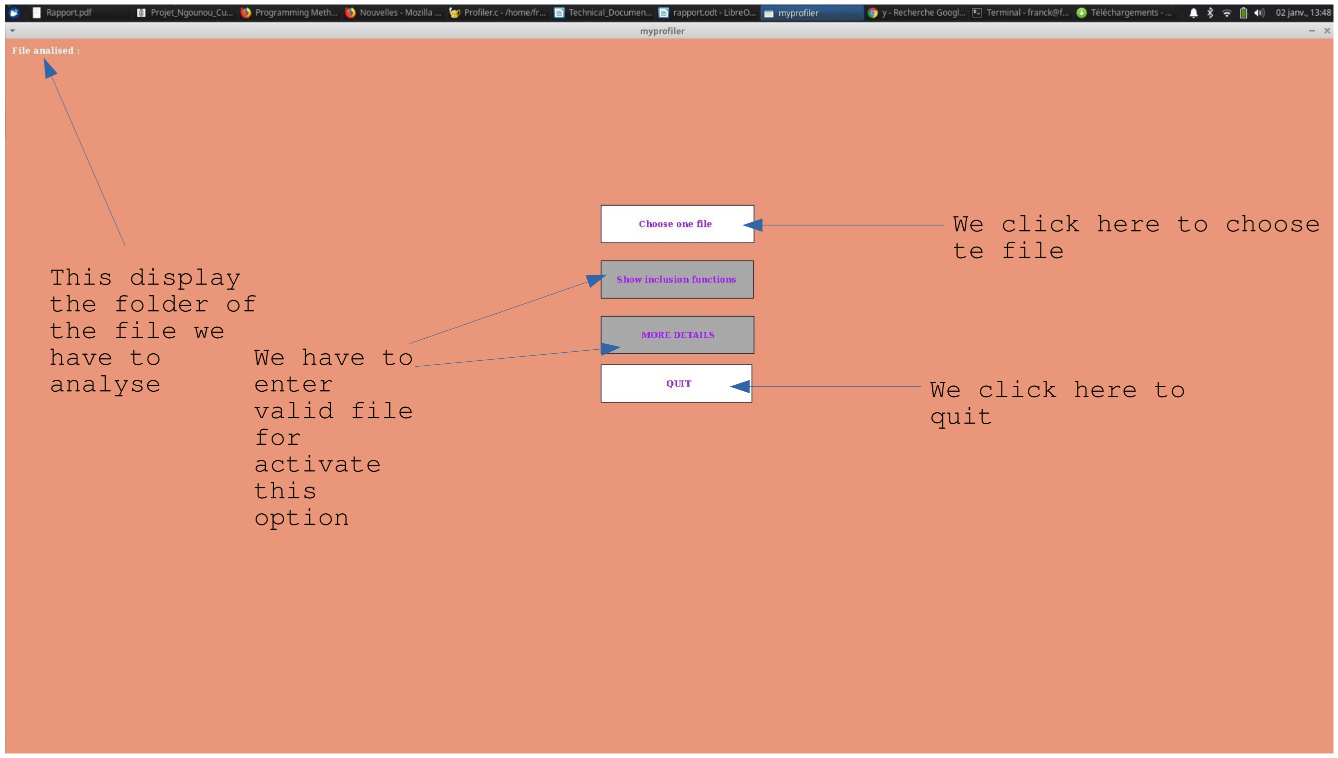
Summary

1. Program Explanation
2. Module Cutting
3. Functions

I) Explanation Program

The goal of this program is to display more precisely the functions executions in details in the optics to have the more optimal program

At first we have the menu with different available option:



This display the folder of the file we have to analyse

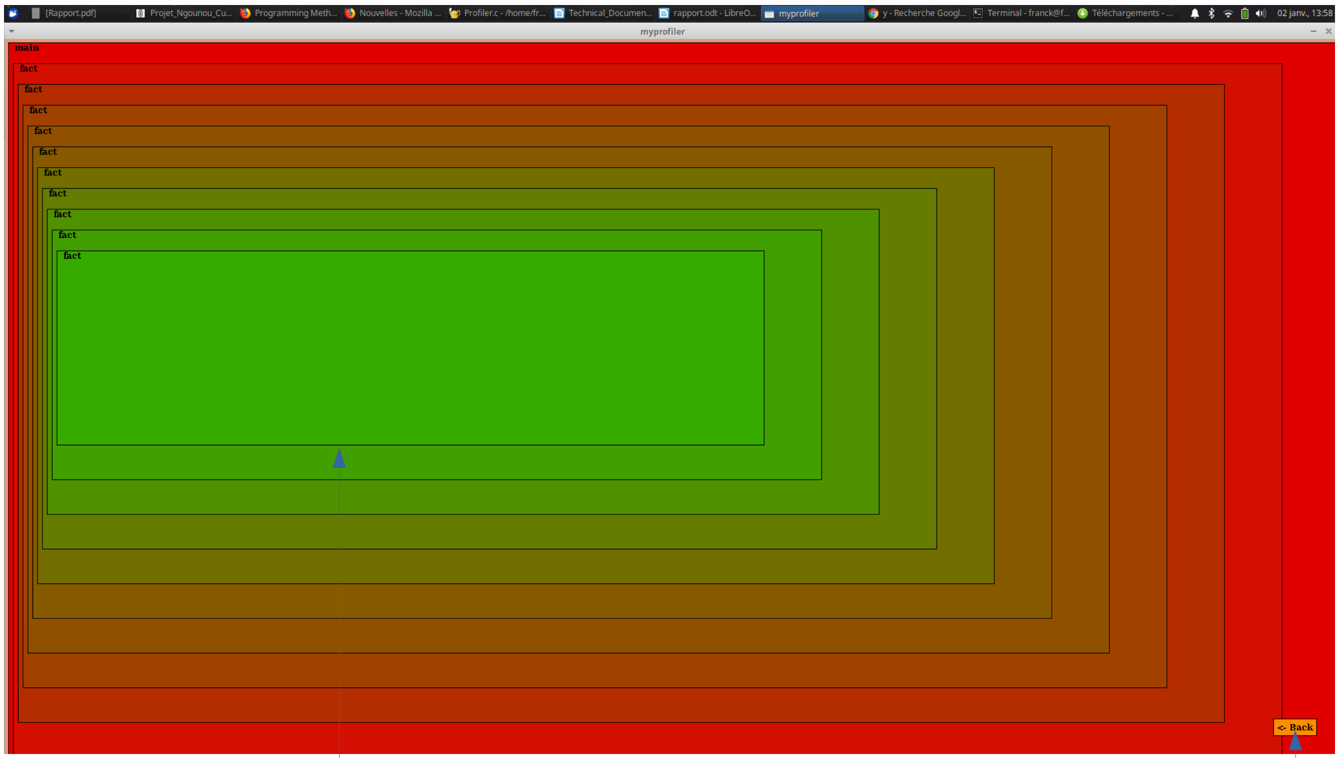
We have to enter valid file for activate this option

We click here to choose te file

We click here to quit

After enter a valid file, we test the different functions:

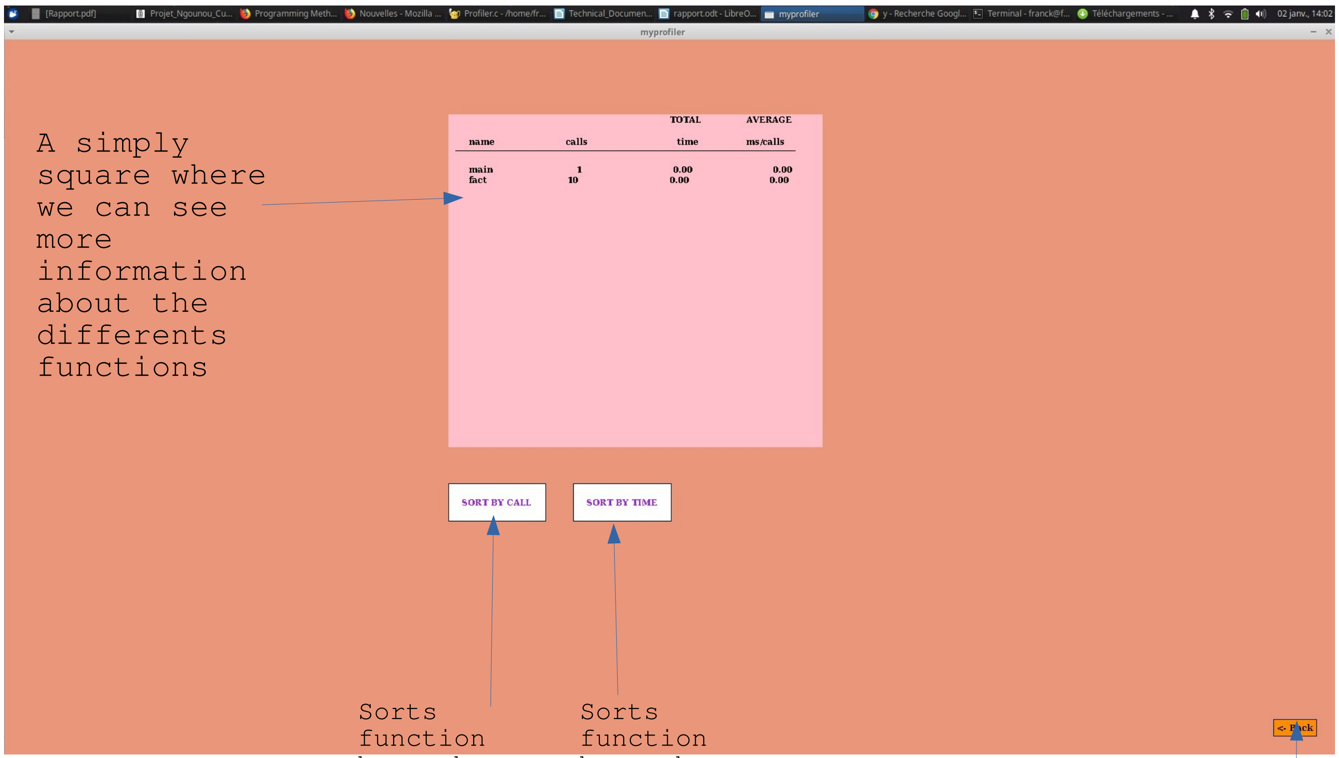
Show function inclusions :



Display function about time
(greener is faster, red
slower)
And where they are called

Button to go back
on general menu

More details



```
                              TOTAL        AVERAGE
 name          calls           time         ms/calls
 ---------------------------------------------------
 main             1            0.00            0.00
 fact            10           0.00            0.00
```

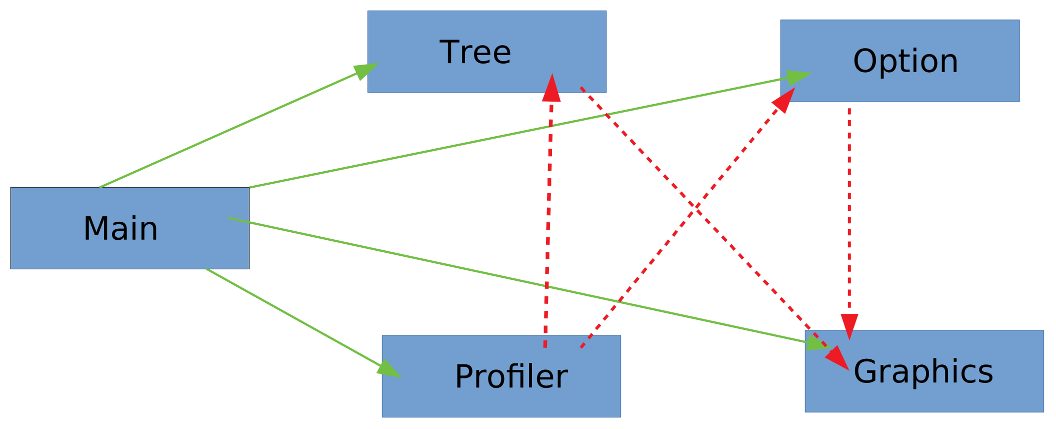
A simply square where we can see more information about the differents functions

Sorts function by number of call

Sorts function by number of time

Button to go back on general menu

II/ Module Cutting



III) Functions

Each function are class by category

III.I/ Function about Tree

```
 static void course_son (Tree a,int nb_brother,int
           which_bro,int nb_son,int height,int width,
        int pos_x,int pos_y,int timer,int father_fx,
        int father_width)
```

This function take a lot of parameters because it take care of lot of differents things. This function course the tree, and in function of the number of brother of each Node, the size of square displaying is adjusted and the number of square on the same line are limited. For each son, we increment the *timer* argument to have greener function each time.

Complexity : ***O(n)***

double *add_function(****Tree**** *a,****FILE***** f)*

This function course the file f, and for each function found, we create a new node with his information.

void *nb_function(****Tree**** *a, ****int**** *i)*

This function how many brother have a the node (*a)

III.II/ Function about Profiler

int *create_profiler(Tree a,****Profiler**** P[])*

This function call 4 annexes function to manage different things
Theses functions are :

- ***int*** *number_function(****Arbre**** a, ****Profiler**** p[],int index)*
  who calculate all the different functions present in the tree.

- ***int*** *number_call(****Arbre**** a,****Profiler**** *p)*
  calculate how many time each functions are call.

- ***int*** *time_per_function(a,&P[i])*
  calculate total time peer function

_ ***int*** *time_per_call(a,&P[i])*
calculate total time peer function call.

Complexity : **O(n)**

```
void profiler_course(Profiler p[],int
                *navigation,int height,int width,
                int option,int number_of_function)
```

This function sort the ‘p’ array in function of the user choice.
(by call,time)

Complexity : **O(1).**