

### TABLE DES MATIERES

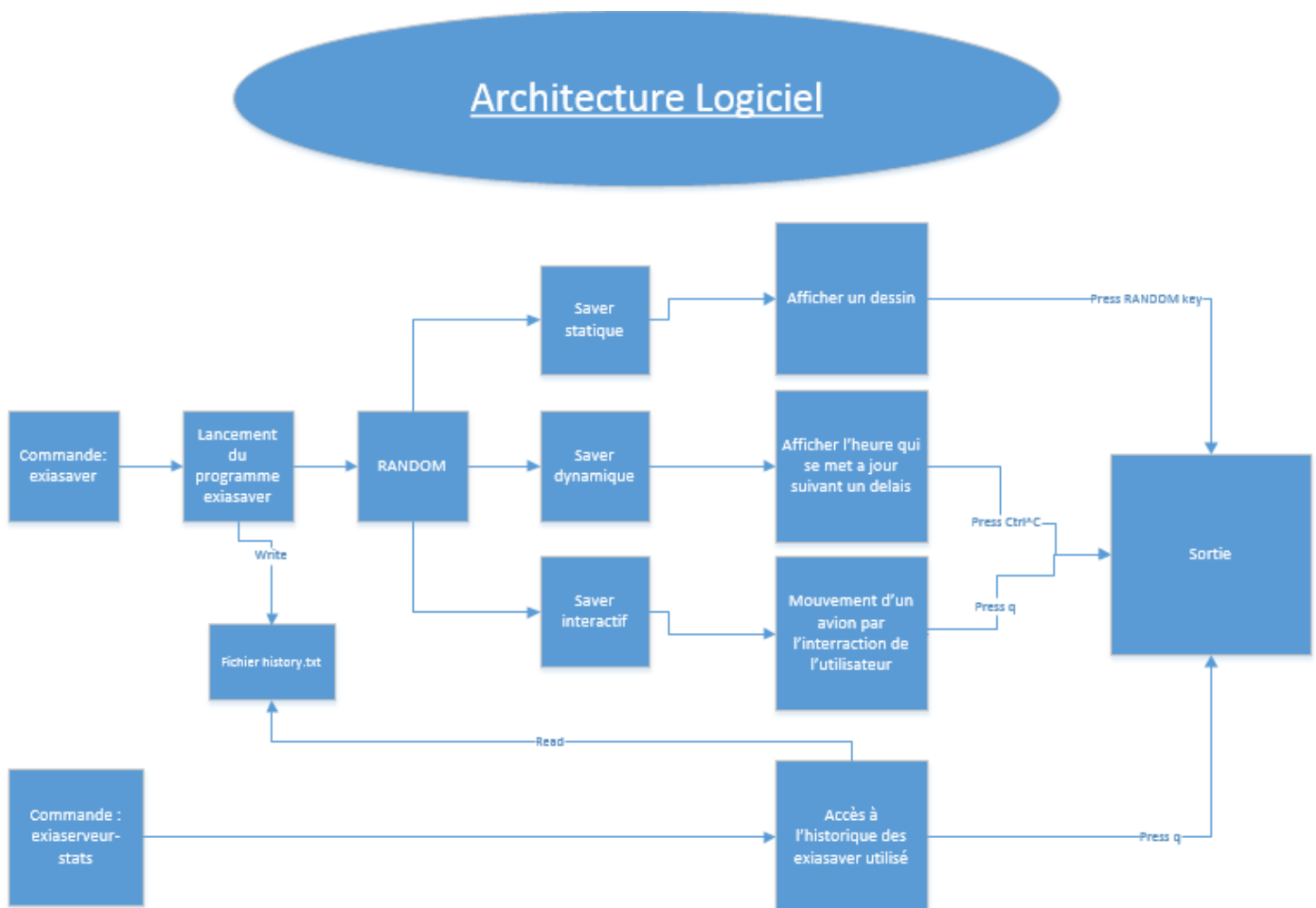
Première partie : ANALYSE.....	3
Deuxième partie : ANALYSE DE DONNEES ET STRUCTURES....	5
LAUNCHER .....	5
SAVER STATIQUE.....	6
SAVER INTERACTIF .....	7
SAVER DYNAMIQUE .....	8
Troisième partie : MODULARISATION & WORKFLOW DE FONCTIONS.....	9
Description graphique de chaque module .....	9
Prototypes de l'ensemble des fonctions du projet .....	13





### PREMIERE PARTIE : ANALYSE

#### 1. Dessinez l'architecture logiciel – comment avez-vous compris le projet ?



## FEUILLE D'AVANCEMENT DU PROJET

Dans ce projet nous avons compris plusieurs choses :

La première est que nous devons produire 3 Ecran de veille dans une console que nous allons présenter lors de notre soutenance.

Nous devons par la suite faire ces 3 screensaver différent

- Saver Statique
- Saver Dynamique
- Saver Interactif

Il faut donc 3 structures différentes, ainsi que des fonctions pour ces screensaver.

Lorsque l'on va exécuter notre commande

– Exia.server, celle-ci va lancer aléatoirement l'un des trois Saver

Le saver sélectionné s'exécutera et l'écran de veille s'affichera



### DEUXIEME PARTIE : ANALYSE DE DONNEES ET STRUCTURES

1. **Représentation graphique de toutes les structures nécessaires. Sans oublier les liens avec les fichiers externes lus ou écrits.**

## LAUCHER

```
//Structure associating a number for each different screenSaver
typedef struct screenSaver screenSaver {

    int statique = 1;
    int dynamique = 2;
    int interactif = 3;

};
```

Structure contenant les différents screensaver, elle sera utilisée pour effectuer une exécution aléatoire



# SAVER STATIQUE

```
//Structure associating a number for each PBM file
typedef struct PBM PBM {

    int chateau = 0;
    int exia = 1;
    int pinguin = 2;
    int poulet = 3;

};
```

Nous avons décidé d'établir une structure de données, constituée de d'entier prenant une valeur de 0 à 3. Elle nous servira pour notre fonction aléa qui se servira des valeurs attribuées afin de tirer aléatoirement un chiffre qui correspondra à un dessin.

Nous avons pris la décision de créer une structure de données de type liste chaînée, qui nous servira pour stocker les bytes que compose les fichiers «.pbm».

On a pris la liberté de d'établir une structure de type liste pour trouver le premier byte contenue dans la liste chaînée.

```
//Linked list containing the PBM's bytes
typedef struct bytes bytes{

    int byte;
    struct bytes *nxt;

};
struct llist {

    struct bytes *first;

};
```



### SAVER INTERACTIF

```
//SCREENSAVER INTERACTIF STRUCTURES ET PROTOTYPES

typedef struct image image {

    *FILE avion_H;
    *FILE avion_B;
    *FILE avion_G;
    *FILE avion_D;

}
```

Nous avons fait le choix de construire une structure pour avoir l'ensemble des images des différentes localisations des avions.

Nous avons créé une structure permettant de stocker la position actuelle et la suivante de l'avion.

```
//save previous and actual plane position
typedef struct avionLoc avionLoc{

    char actual;
    char next ;

}
```

```
//Coordinates for each part of the plane
typedef struct coordoAvion coordoAvion{

    int A1;
    int O1;
    ...
    int A11;
    int O11;

}
```

Nous avons choisi de réaliser une structure nommée « coordoAvion », elle est destinée à contenir les abscisses et les ordonnées de la console sachant que c'est un tableau a deux dimensions de 80 X 23.



## FEUILLE D'AVANCEMENT DU PROJET

# SAVER DYNAMIQUE

```
typedef struct chiffres chiffres {//struct with pointer on number  
  
    *FILE chiffer_un;  
    *FILE chiffer_deux;  
    *FILE chiffer_trois;  
    *FILE chiffer_quatre;  
    *FILE chiffer_cinq;  
    *FILE chiffer_six;  
    *FILE chiffer_sept;  
    *FILE chiffer_huit;  
    *FILE chiffer_neuf;  
};
```

Nous avons créé une structure pour stocker tous les pointeurs vers les fichiers contenant les chiffres.

Nous avons eu l'idée d'une structure contenant la taille des chiffres.

```
typedef struct taillesChiffres taillesChiffres{// size of number  
  
    int x;  
    int y;  
}
```

```
int **tabChiffres;// dynamic tab  
  
tabchiffres= malloc(x*(sizeof (*ptr)));  
tabChiffres[i]= malloc(y * (sizeof(**ptr)));
```

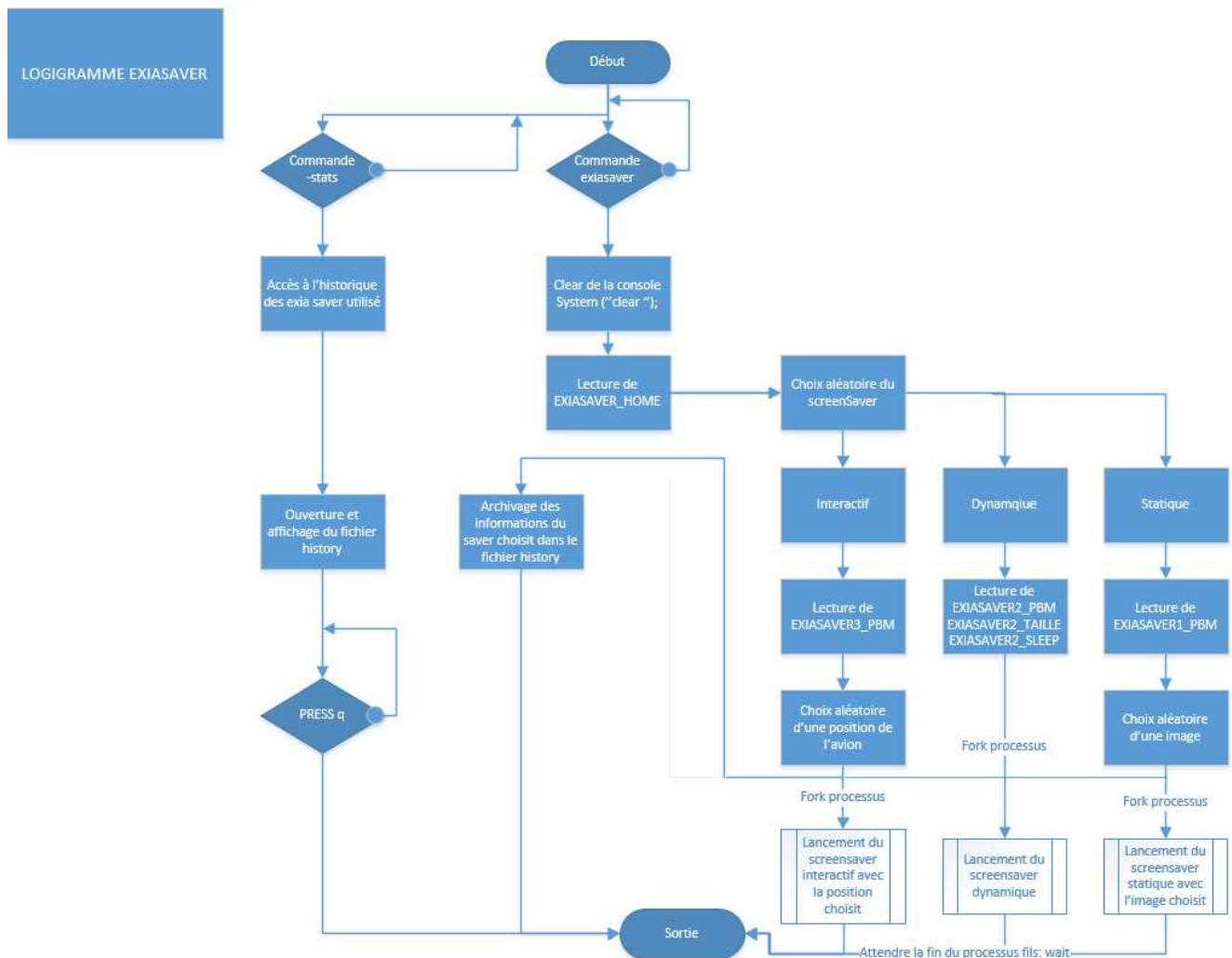
Nous avons souhaité créer un tableau bidimensionnel dynamique qui stockera les bytes des fichiers « pbm » des chiffres.





### TROISIEME PARTIE : MODULARISATION & WORKFLOW DE FONCTIONS

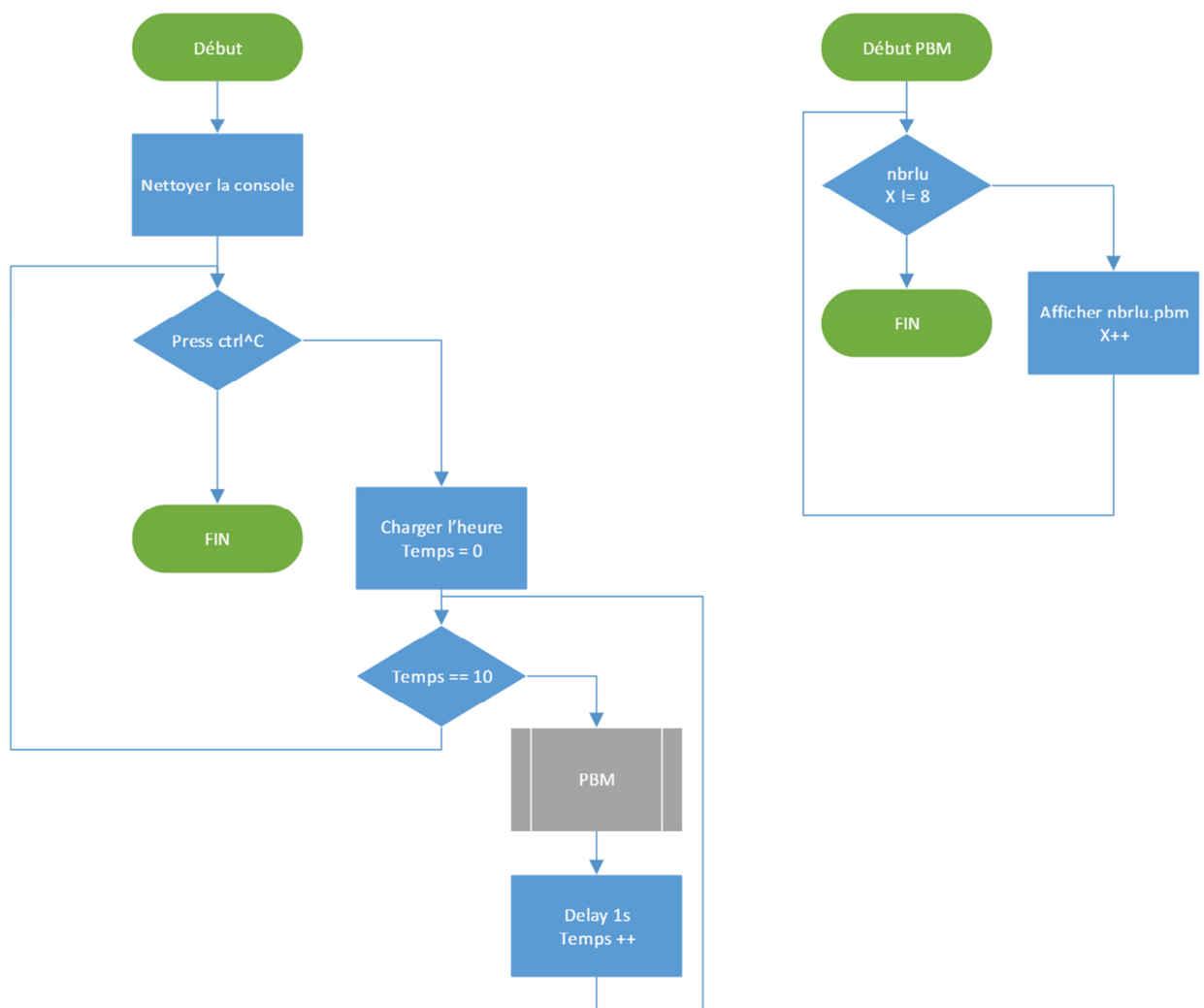
DESCRIPTION GRAPHIQUE CHAQUE MODULE (LANCEUR EXIASAVER ET LES 3 TERMSAVER) - LOGIGRAMME OU WORKFLOW



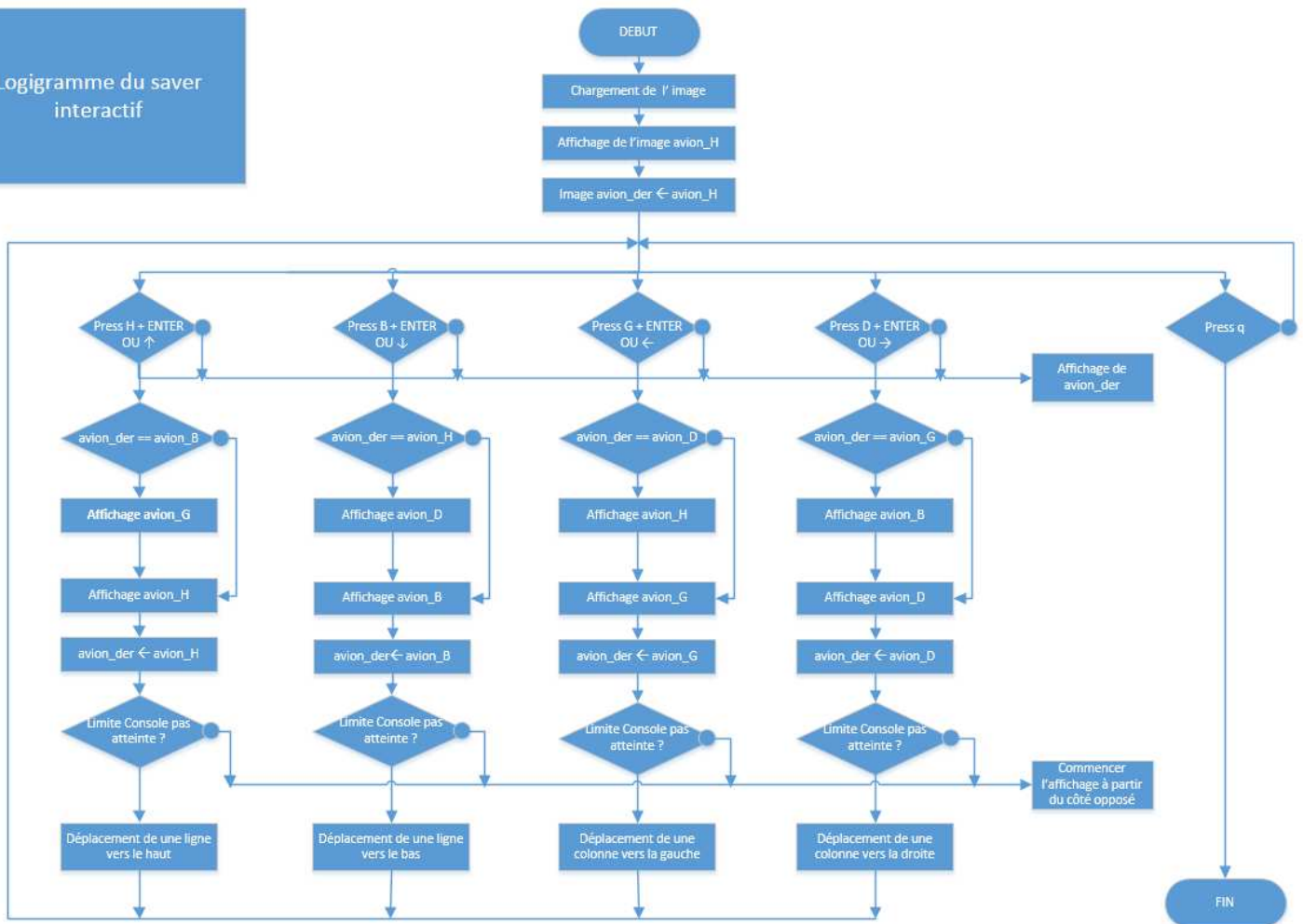
### LOGIGRAMME DU SAVER STATIQUE



### Logigramme du saver dynamique



Logigramme du saver  
interactif



*PROTOTYPES DE L'ENSEMBLE DES FONCTIONS DU PROJET (FAITES-LE LE PLUS « GRAPHIQUE » POSSIBLE)*

## SCREEN SAVER STATIQUE

```
//Open the PBM file  
*FILE ouverture(int image);
```

Fonction permettant d'ouvrir le fichier PPM passé en paramètre lors de l'exécution du programme statique.



```
//Put the PBM's bytes in the linked list  
llist remplissage (*FILE);
```

On lit les bytes du fichier PPM et on les place dans une liste chaînée.



```
//Print the linked list centered on the screen  
void affichage (*llist);
```

On affiche la liste chaînée en la centrant dans la console.



## SCREEN SAVER INTERACTIF

```
//Charge path image and save it in the structure image  
image chargeImage(*image, path_image);
```

Charger l'image dont le chemin est spécifié dans path\_image.

```
//Complete the plane coordinate structure with the bytes  
coordoAvion remplissageAvion(image , *coordoAvion);
```

Lire et écrire les bytes du fichier chargé dans la struct coordoAvion.

```
//Complete the static tab with the coordinate plane structure  
void remplissageTAB(*coordoAvion , *tabConsole);
```

Placer les bytes de l'avion dans le tableau tabConsole en fonction de leurs coordonnées.

```
//Change the coordinates of the planes for move it  
int deplacement(*avionLoc, *coordoAvion);
```

Changement des coordonnées de l'avion pour le déplacer.

```
//Print to the screen tabConsole containing the plane  
void affichage(*tabConsole);
```

Afficher le tableau tabConsole à l'écran.



### SCREENSAVER DYNAMIQUE :

```
//Read the time and store it into a string char  
char readTime ();
```



Récupérer l'heure dans une chaîne de caractère.

```
//Load a number  
void charge(*chiffres);
```



Charger une image représentant un chiffre

```
//Complete the PPM's bytes in tabChiffres  
void remplissageTab(chiffres , *tabChiffres);
```



Charger une image représentant un chiffre

```
//Complete the tabConsole with tabChiffres  
void remplissageTabConsole(*tabChiffre , *tabConsole);
```

Place les bytes tabChiffres dans tab Console



```
//Print the time  
void affichage (*tabConsole);
```

Afficher le tabConsole



### EXIA LAUNCHER

```
//Random choice between the different PBM files numbers  
int aleaStruct(struct);
```

Choix aléatoire : image avion,  
type de screenSaver



```
//Open fichier history.txt  
FILE openHistory(char path);
```

Ouvrir le fichier history.txt



```
//Save informations about screenSaver launched into history.:  
FILE addInfo(FILE*, int time, int date, int type);
```

Ecrire les informations d'historique dans le fichier history.txt





# PROJET – PROGRAMMATION SYSTEME

## FEUILLE D'AVANCEMENT DU PROJET

### Quatrième partie : REPARTITION DES TACHES

**Nom :** Van-Camp Rémy

**Rôle principal :** Membre de projet

Tâches	7/12	8/12	9/12	12/12	13/12	14/12	15/12	16/12
Conception	Architecture logiciel	Logigramme Interactif Structure Fonction	Structure Fonction					
Dev			Codage Interactif	Codage Interactif	Codage Interactif	Test Manuel Amélioration		
Management							PPT Rapport De projet	ORAL



# PROJET – PROGRAMMATION SYSTEME

## FEUILLE D'AVANCEMENT DU PROJET

Nom : Rigaut Arnaud

Rôle principal : Membre de projet

Tâches	7/12	8/12	9/12	12/12	13/12	14/12	15/12	16/12
Conception	Analyse donnée et structure (compréhension)	Logigramme Dynamique Structure Fonction	Structure Fonction					
Dev			Codage Statique	Codage Dynamique	Codage Dynamique	Test Manuel Amélioration		
Management							PPT Rapport De projet	ORAL



# PROJET – PROGRAMMATION SYSTEME

## FEUILLE D'AVANCEMENT DU PROJET

**Nom :** Alexandre Lecompte

**Rôle principal :** Membre de projet

Tâches	7/12	8/12	9/12	12/12	13/12	14/12	15/12	16/12
Conception	Analyse donnée et structure (compréhension)	Logigramme Dynamique Structure Fonction	Structure Fonction					
DEV			Codage Interactif	Codage interactif	Codage interactif	Test Manuel Amélioration		
Management							PPT Rapport De projet	Oral



# PROJET – PROGRAMMATION SYSTEME

## FEUILLE D'AVANCEMENT DU PROJET

**Nom :** Podevin Jean clément

**Rôle principal :** Chef de projet

Tâches	7/12	8/12	9/12	12/12	13/12	14/12	15/12	16/12
Conception	Git Hub, MS Project, planning	Logigramme Statique  MS Project – Feuille avancement	Feuille Avancement					
Dev			Codage Statique	Codage Dynamique	Codage Dynamique	Test Manuel  Amélioration		
Management							PPT  Rapport  De projet	ORAL

