BESSA Alexandre :
alexandrebessa26@gmail.com
ELBEZ Samuel :
samuel.elbe@gmail.com
JACQUETTE Pierrick :
pierrick.jacquette@gmail.com

TP n°8 du 02/12/2017
Disk B Tree
21306128
21200353
21305551

TP8

Cahier des charges

On doit réaliser un B+Tree sur disque.
Pour simplifier on part de fichiers (modélisant les blocks) qui sont triés.
Il y aura un makefile et le langage utilisé est le C.
Une documentation sous forme de man pages est demandé.
La librairie fourni doit disposer de quatres fonctionnalités : créer un arbre, insérer dans un arbre, pouvoir stocker en disque un arbre et pouvoir charger un arbre en mémoire.

Dossier de conception

C'est une application Linux ou Mac
La date de livraison sera le 08 décembre 2017.
L'application permettra de persister en disque un B Tree afin de pouvoir s'en resservir ultérieurement, par ailleurs les feuilles contiendront des références (en mémoire) vers le contenu en disque.

Structure :

Nous avons commencé par réfléchir au aspect techniques du tp. De quel information avons nous besoin de stocker, éviter de parcourir l'arbre pour connaître le chaînage des nœuds, autant la stocker pour chaque nœud. Nous avons donc opter pour cette structure :

```
typedef struct {
        int isLeaf;
        short value1, value2;
        short s1, s2, s3;
        struct Node *left, * middle, *right;
        struct Node *father;
        short level, indic;
} Node;
```

Architecture :

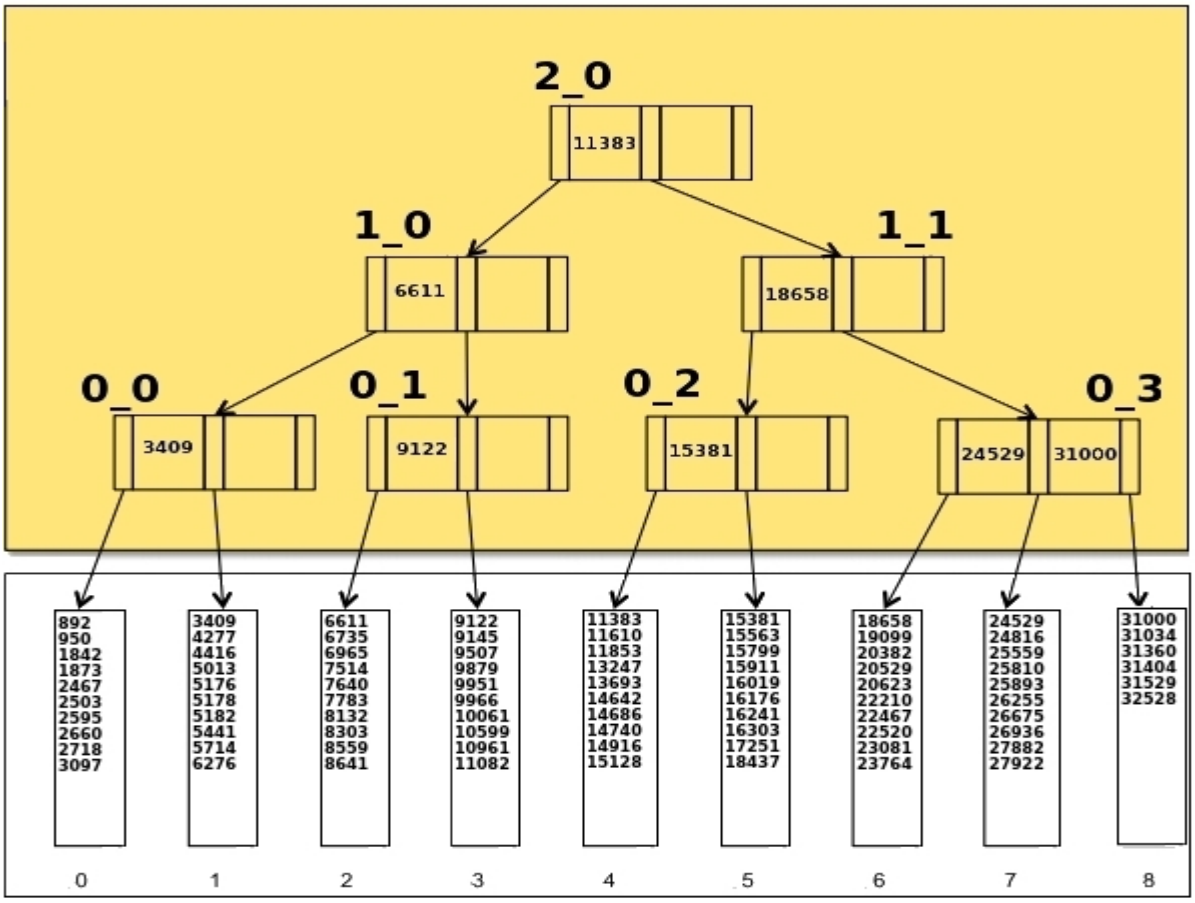
- config.h : contient les macro d'erreurs
- diskIO (.h et .c) : permet de lire un fichier (et le stocker dans un tableau), d'écrire le contenu d'un tableau dans un fichier.
- LibBT (.h et .c) : contient les fonctions pour gérer l'arbre B Tree.
- relation (.h et .c) : contient les fonctions pour gérer et générer les répertoires.
- main.c : permettant de faire les appels des autres fonctions en fonction du tp.

BESSA Alexandre :
alexandrebessa26@gmail.com
ELBEZ Samuel :
samuel.elbe@gmail.com
JACQUETTE Pierrick :
pierrick.jacquette@gmail.com

TP n°8 du 02/12/2017
Disk B Tree
21306128
21200353
21305551

Choix d'implémentation

Les fichiers contiendront des shorts strictement positifs, pour persister les données nous avons choisi pour chaque nœud de créer un fichier ayant comme nom sa hauteur (en partant des feuilles) dans l'arbre ainsi que son indice. Par exemple :



Dans chaque fichier en disque nous avons choisi ce format :

Ligne 1 : nom du fichier contenant le pére du nœud courant
Ligne 2 : valeur de la case 1 (-1 si pas de valeur)
Ligne 3 : valeur de la case 2
Ligne 4 : est-ce une feuille ? 1 si oui, sinon 0
Ligne 5 : nom du fichier S contenu à gauche
Ligne 6 : nom du fichier S contenu au milieu
Ligne 7 : nom du fichier S contenu à droite

BESSA Alexandre :
alexandrebessa26@gmail.com
ELBEZ Samuel :
samuel.elbe@gmail.com
JACQUETTE Pierrick :
pierrick.jacquette@gmail.com

TP n°8 du 02/12/2017
Disk B Tree
21306128
21200353
21305551

Man pages

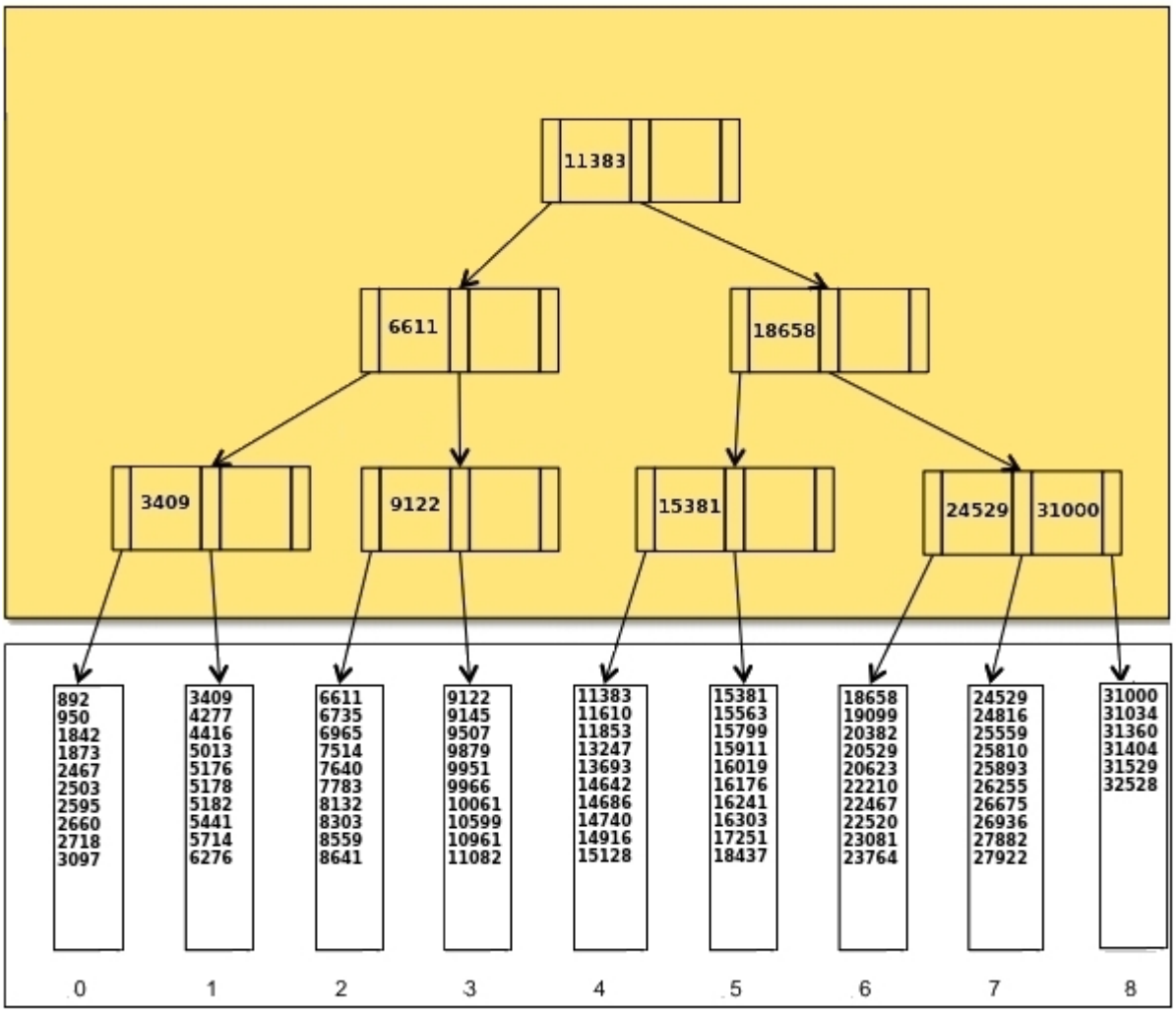
Des pages de man qui sont consultables depuis un terminal sont présentes dans le rendu. La documentation sous forme d'un site internet est disponible à cette adresse : https://www.dropbox.com/sh/oy0x97ykps4db6n/AABvS6Sb9vcLWcAhZmCWfevga?dl=1

Run :

Pour compiler le projet faire make
Pour exécuter faire make run
Pour visualiser une man pages : man -l XXX

Exemple :

Avec nos fichiers S d'exemples cela crée l'arbre suivant :



BESSA Alexandre :
alexandrebessa26@gmail.com
ELBEZ Samuel :
samuel.elbe@gmail.com
JACQUETTE Pierrick :
pierrick.jacquette@gmail.com

TP n°8 du 02/12/2017
Disk B Tree
21306128
21200353
21305551

Difficulté :

La version de l'arbre en mémoire nous a posé des problèmes techniques avec les pointeurs du tableau. Nous avons finalisé une version en disque où nous avions plus besoin des pointeurs comme l'on stocke uniquement une référence et non pas tout le fichier. Le but du client étant la version disque : la version mémoire étant en tremplin afin d'effectuer les opérations sur disques.