Evaluation

# Préambule

## Précaution :

* Respecter l’indentation de vos programmes
* Commenter bien vos programmes

## Organisation du projet

Créer un répertoire Evaluation

Vous compresserez votre projet en tar.gz puis vous enverrez l’archive compressée par mail à l’adresse suivante : [ludovic.drouineau@isen-ouest.yncrea.fr](mailto:ludovic.drouineau@isen-ouest.yncrea.fr)

Rappel

tar cvf Evaluation.tar Evaluation

gzip Evaluation.tar

# Les arbres binaires :

Ecrire dans un fichier arbre.c, la fonction main qui fera appelle aux fonctions ci-dessous. Les fonctions seront codées dans un fichier (arbre\_util.c). Le fichier arbre.c fera appel aux fonctions codées dans arbre\_util.c et utilisera le fichier arbre\_util.h.

Créer un makefile.

Soit la structure suivante :

struct node {

int val ;

struct node\* left ;

struct node\* right ;

}

### Fonction newNode:

struct node\* newNode(int value)

Créer une fonction qui prend un entier en paramètre et créée un nouveau noeud. Elle renverra le nouveau nœud.

### Fonction insertNode

struct node\* insert(struct node\* node, int key)

Créer une fonction insertNode qui prend un nœud en paramètre, une clé et qui créé un nœud avec la nouvelle valeur soit à gauche si la valeur est plus petite, soit à droite. Cette fonction utilisera la fonction précédente pour créer un nouveau nœud.

Exemple :

Soit le nœud suivant :

Si on insère 110, l’arbre deviendra :

Si on insère 90 :

### Fonction releaseNode

void releaseNode(struct node\* root)

Ecrire une fonction qui libère toute la mémoire allouée pour un nœud donné (récusirvement elle libérera de la mémoire pour le fils gauche et le fils droit)

### Fonction searchNode

struct node\* search(struct node\* root, int key)

Ecrire une function searchNode qui prend en parameter un noeud et renvoie le noeud don’t la valeur est égale au parameter passé.

### Fonction printOrder

void printOrder(struct node\* root)

Ecrire une fonction printOrder qui prend en paramètre un arbre et affiche les valeurs de cet arbre du plus petit au plus grand.

### Test des fonctions

Créer un main qui va insérer successivement les valeurs suivantes :

100 ; 80 ; 120 ; 90 ; 70 ; 110 ; 130.

Vérifier que tout fonctionne.

# Les chaines de caractères

On écrira les fonctions suivantes dans un seul fichier chaine.c qui contiendra également le main.

### Longueur d’une chaine

int stringLength(char\* chaine)

Ecrire une fonction stringLength qui retourne la longueur d’une chaine de caractère sans utiliser la fonction strlen.

## Majuscule / minuscule

Ecrire une fonction qui prend une chaine de caractère rentrée par l’utilisateur ;

* si cette chaine de caractère est en minuscule : la fonction retourne une chaine de caractère en majuscule ;
* à l’inverse si celle-ci est en majuscule : la fonction retourne la chaine en minuscule.