Objectif : créer une bibliothèque de fonctions sur les arbres binaires.
Testez les fonctions au fur et à mesure via un programme principal.

1. Création d'un arbre binaire

Question 1 *Créez un fichier* `arbrebinaire.h` *dans lequel vous déclarerez un type arbre qui sera un pointeur vers une structure noeud, un nœud étant composé d'un élément et de 2 pointeurs vers des nœuds (sous-arbre gauche et sous-arbre droit). Pour ce TP, nous considèrerons des arbres contenant des entiers.*

Question 2 *Déclarez les fonctions suivantes que vous implanterez dans le fichier* `arbrebinaire.c` *:*

```
arbre ArbreVide(); // renvoie un arbre vide
int EstArbreVide(arbre ); // teste si un arbre est vide

element Racine(arbre); //renvoie la valeur de la racine de l'arbre
arbre Gauche(arbre); //renvoie le sous arbre gauche
arbre Droit(arbre);//renvoie le sous arbre droit

arbre Construire(element, arbre, arbre); // construit un arbre
      a partir de sa racine et de 2 sous arbres
```

Testez vos fonctions à l'aide d'un programme principal `main.c`.

2. Les parcours d'arbres binaires

Question 3 *Ajoutez à la bibliothèque de fonctions précédemment créée, les fonctions permettant de faire les 3 parcours vus en cours :*

```
void ParcoursPref(arbre);
void ParcoursInf(arbre);
void ParcoursSuff(arbre);
```

Question 4 *Utiliser ces fonctions dans un programme principal pour initialiser un arbre et afficher son contenu.*

3. Les arbres binaires de recherche

Question 5 *Implantez les fonctions permettant d'insérer et de rechercher un élément dans un abr :*

```
arbre Insertion(element, arbre );
int Recherche (element e, arbre A);
```

Autres fonctions sur les arbres binaires de recherche

Question 6 *Implantez les fonctions permettant de connaitre le plus grand élément contenu dans un abr et les fonctions permettant d'en supprimer un élément :*

```
element Max(arbre); /*retourne l'element max d'un arbre, utiliser dans SuppressionRacine*/

arbre SuppressionMax(arbre);
arbre SuppressionRacine (arbre);
arbre Suppression(element, arbre);
```

Question 7 *Utilisez ces fonctions dans un programme principal.*

4. Autres fonctions sur les arbres

Pour aller plus loin, vous pouvez implanter les fonctions suivantes :

```
int Hauteur(arbre);
int NbNoeud(arbre);
```

et le *tri par tas* (vu en cours).

Il est intéressant également de réfléchir à une structure pour représenter les arbres n-aire, d'implanter les fonctions permettant de faire une recherche dans un arbre n-aire et de convertir un arbre n-aire en arbre binaire.