QROC A31 Structure de données Janvier 1999

Documents distribués et notes personnelles autorisés.

Il faut impérativement répondre sur la copie-sujet.

Faire d'abord au brouillon, aucun double de sujet ne pourra être distribué.

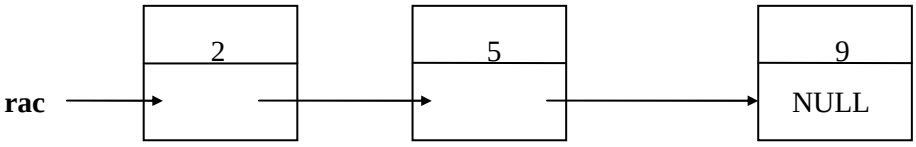
Nom :

Prenom :

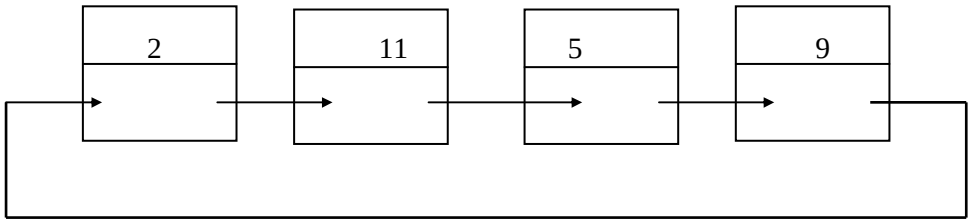
Liste chaînée

- **Exercice 1**

On considère la structure chaînée initiale suivante :



Où rac désigne le pointeur de début de cette liste. On veut créer la liste suivante :



Parmi les cinq portions de codes proposées une seule réalise cette transformation (répondre dans le tableau réponse). Toutes ont en commun les déclarations suivantes :

```
typedef struct list_noeud *LISTE;
typedef struct list_noeud {
        int valeur;
        struct list_noeud *suiv;
        };
LISTE rac;

/*********************Proposition  1*********************/
LISTE t;
t=(LISTE)malloc(sizeof(list_noeud));
rac->suiv=t;
t->suiv=rac->suiv->suiv->suiv;
t->valeur=11;
```

```
/*******************Proposition  2 ************************/
LISTE t;
rac=rac->suiv->suiv;
t=(LISTE)malloc(sizeof(list_noeud));
t->suiv=rac;
t->valeur=11;
rac=t;
/************************proposition 3************************/
LISTE t;
t=(LISTE)malloc(sizeof(list_noeud));
t->suiv=rac;
t->valeur=11;
rac=t;
/****************proposition 4*************************/
LISTE t;
t=(LISTE)malloc(sizeof(list_noeud));
t->suiv=rac->suiv;
t->valeur=11;
rac=t;
/**********************proposition  5**********************/
LISTE t;
t=(LISTE)malloc(sizeof(list_noeud));
t->suiv=rac->suiv;
t->valeur=11;
rac->suiv=t;
rac->suiv->suiv->suiv->suiv=rac ;
```

Réponse :

Recopie des lignes	*Schémas correspondants*

Exercice 2

Dans cet exercice nous souhaitons être capable de représenter un nombre quelconque de polynômes différents tant qu'il y a de la mémoire disponible. En général, nous voulons représenter le polynôme :

$$A(x) = a_{m-1}x^{e_{m-1}} + a_{m-2}x^{e_{m-2}} + ... + a_0 x^{e_0}$$

où les a_i sont des coefficients réels non nuls et les e_i sont des exposants entiers positifs tels que $e_{m-1} > e_{m-2} > .... > e_1 > e_0 >= 0$.

Nous souhaitons représenter les polynômes sous forme de listes simplement chaînées. Une structure adéquate pour représenter un polynôme peut être définie de la façon suivante :

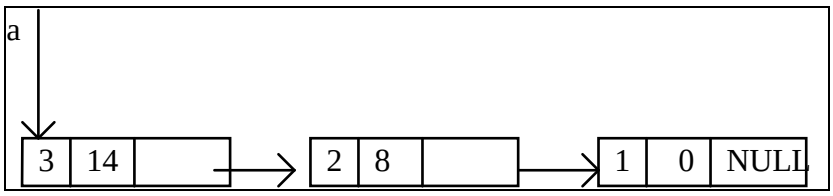
```
typedef struct noeud *poly;

    struct noeud {
            float coefficient;
            float exposant;
            poly lien;
    };
```

où les champs coefficient et exposant désignent respectivement les coefficients et les exposants du polynôme.

Exemple :

Soit le polynôme $a(x)=3x^{14}+2x^8+1$

Ce polynôme peut être représenté sous la forme suivante :



1) Ecrire une fonction **poly integral(poly p)** qui permet de calculer l'intégrale d'un polynôme.
Exemple : Soit le polynôme $a(x)=3x^{14}+2x^8+1$

intergale(a) retourne $b(x)=3/15x^{15}+2/9x^9+x$

Rappel : l'intégrale du monôme x^n est $\frac{1}{n+1}x^{n+1}$.

Réponse :

2) Sachant que deux polynômes sont identiques si et seulement si ils ont les mêmes coefficients et les même exposants, écrire une fonction **int Egal(poly p, poly q)** qui permet de tester si deux polynômes sont identiques. Egal retourne 1 si les deux polynômes sont identiques, 0 sinon.

Exemple1 : Soit le polynôme $a(x)=3x^{14}+2x^{8}+1$, soit $q(x)= a(x)=3x^{14}+2x^{8}+1$, les polynômes p et q sont identiques. Egal retourne 1.

Exemple2 : Soit le polynôme $a(x)=3x^{14}+2x^{8}+1$, soit $q(x)= a(x)=3x^{2}+2x^{8}+1$, les polynômes p et q sont différents. Egal retourne 0.

Réponse :

3) Ecrire une fonction qui réalise la multiplication d'un polynôme par un scalaire **poly Multiplication(poly p, float x)**.

.

Exemple : 2*a(x) = $\mathbf{6x^{14}+4x^8+2}$

Réponse :

4) On s'intéresse dans cet exercice à l'addition de deux monômes :

Exemples:

Soient les monômes a(x) et b(x) définis par : a(x) = $\mathbf{3x^{14}}$, b(x) = $\mathbf{2x^{14}}$ **,** a(x)+b(x)=$\mathbf{5\ x^{14}}$

Soient les monômes a(x) et b(x) définis par : a(x) = $\mathbf{3x^4}$, b(x) = $\mathbf{2x^5}$ **,** a(x)+b(x)=$\mathbf{3x^4+2x^5}$

Soient les monômes a(x) et b(x) définis par : a(x) = $\mathbf{3x^8}$, b(x) = $\mathbf{2x^2}$**,** a(x)+b(x)=$\mathbf{2x^2+3x^8}$

Ecrire une fonction qui réalise l'addition de deux monômes **poly AdditionMonome(poly p, poly q)**.

Réponse :

5) Ecrire une fonction qui permet d'additionner deux polynômes **poly AdditionPolynome(poly p, poly q)**.

Réponse :

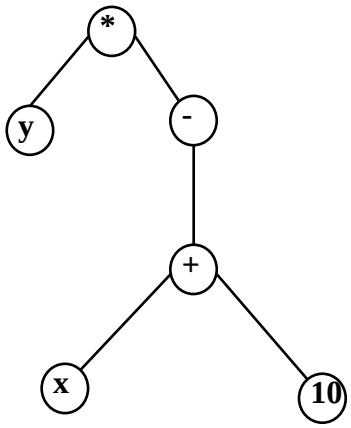
ATTENTION : On ne vous demande pas de créer les listes chaînées, on suppose qu'elles existent.

Arbres

Exercice 1

Une expression arithmétique peut être facilement représentée par un arbre. Les arbres d'expression spécifient l'association des opérandes d'une expression et de ses opérateurs d'une manière uniforme, sans qu'il soit nécessaire de se préoccuper du placement des parenthèses.

Exemple : l'expression suivante (y*(-(x+10))) peut être représentée par l'arbre



1) Construisez l'arbre de l'expression suivante : ((x+y)+(5*(x+z)))

Réponse :

2) Donner une structure de données possible pour représenter les arbres définis précédemment.

Réponse :

3) Dans cette partie nous ne considérons pas comment l'arbre a été crée mais nous supposons qu'il existe. Soit la fonction définie par :

```
void affichegdr(tree_pointer ptr){
if (ptr) {
        mystere(ptr->fils_gauche);
        printf("%c",ptr->data);
        mystere(ptr->fils_droit);
        }
}
```

a) En appliquant AfficheGdr sur l'arbre défini dans la question 1, donner le résultat d'affichage de cette fonction

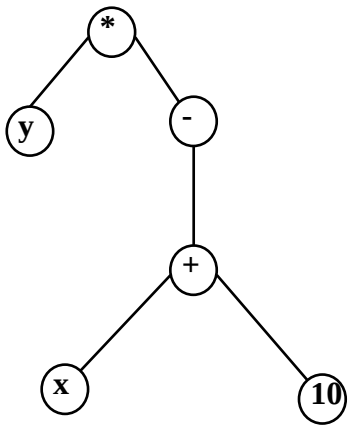
Réponse :

b) Ecrire l'expression ((a+b)+c*(d+e)+f)*(g+h) sous forme préfixée et postfixée.

Réponse :

4) Ecrire une fonction **AfficheGdrParenthese(tree_pointer ptr)** qui retourne l'expression arithmétique avec des parenthèses .

En appliquant la fonction **AfficheGdrParenthese(tree_pointer ptr)** sur l'arbre suivant :



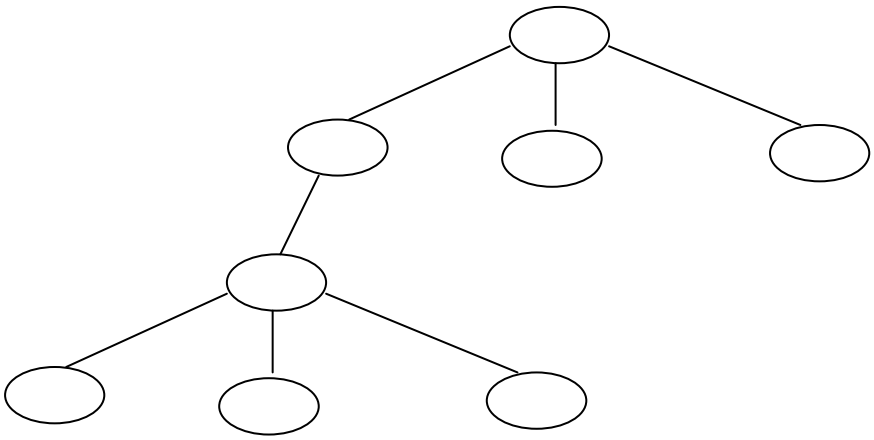
elle devrait retourner (y*(-(x+10))).

Réponse :

Exercice 3 :

Un arbre ternaire (AT) est un arbre tel que chaque père a 3 sous arbres fils. Un sous arbre gauche (SAG), un sous arbre droit SAD et un sous arbre milieu SAM.

Voici un exemple de sous arbre ternaire :



La structure de tel arbre peut être définie de la façon suivante:

```
typedef struct noeud *ATptr;
        typedef struct noeud {
                int valeur;
                ATptr SAG,SAD, SAM ;
        };
```

a) La fonction qui calcule le minimum des valeurs des nœuds d’un arbre ternaire.

Réponse