TD 8 - Listes

Polynômes

Soit le type polynôme défini comme suit :

```
Type poly = ^terme
     terme =    Enregistrement
                     coef : réel
                     degré : entier
                     suiv : poly
                FinEnregistrement
```

1. Ecrire en pseudo-langage une procédure qui réalise la création d'un polynôme.

Ajout en queue de liste

```
Procédure créer-poly-q (S p : poly)
Var q : poly
    i,n,d : entier
    c : réel
Début
  p←nil
  écrire ('entrez le nb de termes')
  lire(n)
  Pour i←1 à n inc +1 Faire
    écrire ('Entrez le coef et le degré >')
    lire(c)
    lire(d)
    Si p =nil (* cas où i=1 *)
       Alors p←allouer(terme)
             p^.coef←c
             p^.degré←d
             q←p
       Sinon q^.suiv←allouer(terme)
             q←q^.suiv
             q^.coef←c
             q^.degré←d
    FinSi
  FinPour
  q^.suiv←nil
Fin
```

Ajout en tête de liste

```
Procédure créer-poly-t (S p : poly)
Var q : poly
    i,n : entier
Début
  p←nil
  écrire ('entrez le nb de termes')
  lire(n)
  Pour i←1 à n inc +1 Faire
    q←allouer(terme)
    écrire ('Entrez le coef et le degré <')
    lire(q^.coef)
    lire(q^.degré)
    q^.suiv←p
    p←q
  FinPour
Fin
```

2. Ecrire en pseudo-langage une fonction qui réalise le calcule de p en x.

3. Ecrire en pseudo-langage une fonction qui réalise l'addition.

```
Fonction add-poly (p1, p2 : poly) : poly
Var p3,q1,q2,q3,t,res : poly
Début
p3←nil
q1←p1
q2←p2
TantQue q1≠nil et q2≠nil Faire
  Si [(q1^.degré=q2^.degré) et (q1^.coef+q2^.coef=0)]
     Alors q1←q1^.suiv
           q2←q2^.suiv
     Sinon t←allouer(terme)
           Si q1^.degré=q2^.degré
                 Alors t^.coef←q1^.coef+q2^.coef
                       t^.degré←q1^.degré
                       q1←q1^.suiv
                       q2←q2^.suiv
                 Sinon Si q1^.degré < q2^.degré
                            Alors t^.degré←q1^.degré
                                  t^.coef←q1^.coef
                                  q1←q1^.suiv
                            Sinon t^.degré←q2^.degré
                                  t^.coef←q2^.coef
                                  q2←q2^.suiv
                       FinSi
            FinSi
            Si p3=nil
                 Alors p3←t
                 Sinon q3^.suiv←t
            FinSi
            q3←t
  FinSi
FinTantQue
Si q1=nil
    Alors res←q2
    Sinon res←q1
FinSi
TantQue res≠nil Faire
       t←allouer(terme)
       t^.degré←res^.degré
       t^.coef←res^.coef
       res←res^.suiv
       Si p3=nil
            Alors p3←t
                  q3←p3
            Sinon q3^.suiv←t
                  q3←q3^.suiv
       FinSi
FinTantQue
Si p3≠nil
  Alors q3^.suiv←nil
FinSi
Retourner(p3)
Fin
```

4. Ecrire en pseudo-langage une fonction qui réalise la multiplication

```
Fonction prod-poly (p, q : poly) : poly
Var p1,q1,temp,r, res : poly
Début
p1←p
res←nil
TantQue p1≠nil Faire
    q1←q
    temp←nil
    TantQue q1≠nil Faire
       Si temp=nil
            Alors temp←allouer(terme)
                  r←temp
            Sinon r^.suiv←allouer(terme)
                  r←r^.suiv
       FinSi
       r^.degré←p1^.degré+q1^.degré
       r^.coef←p1^.coef*q1^.coef
       q1←q1^.suiv
    FintTantQue
    r^.suiv←nil
    r←add-poly(res,temp)
    TantQue temp≠nil Faire  (* récupération des termes de temp *)
       Supprimer(temp,1)
    FinTantQue
    TantQue res≠nil Faire  (* récupération des termes de res *)
       Supprimer(res,1)
    FinTantQue
    res←r
    p1←p1^.suiv
FinTantque
Retourner(res)
Fin
```