Type nœud en Enregistrement

Champ opérateur en Caractère

Champ valeur en Réel

Champ gauche, droit en pointeur de nœud

Fin d’enregistrement

Car == ’ ‘

Ecrire(‘Entrez une expression postfixée’)

Initialiser\_pile()

Tant que (NON fin de ligne()) Faire

Res = 0

Lire(car)

Si (NON fin de ligne) Alors

Nouveau\_noeud = allouer(noeud)

(\*Nouveau\_noeud).gauche = Nil

(\*Nouveau\_noeud).droite = Nil

Si ((car >= ’0’) ET (car <= ‘9’)) Alors

Tant que ((car >= ’0’) ET (car <= ‘9’)) Alors

Chiffre = code accii( car) – 48

Res = ((res \* 10) + chiffre)

Lire(car)

FinTantQue

(\*Nouveau\_noeud).opérateur = ‘0’

(\*Nouveau\_noeud).valeur = res

Empiler(Nouveau\_noeud)

Sinon Si((car = ‘+’) OU (car = ‘-’) OU (car = ‘\*’) OU (car = ‘/’)) Alors

(\*Nouveau\_noeud).opérateur = car

(\*Nouveau\_noeud).valeur = 0

(\*Nouveau\_noeud).gauche = dépiler()

(\*Nouveau\_noeud).droite = dépiler()

FinSi

FinSi

FinTantQue

Racine\_arbre = dépiler()

Foncction évaluer(pt\_noeud)

Déclarations

Paramètre pt\_noeud en poinyeur de nœud

Variables valeur1, valeur2 en réels

Début

Si (($pt\_noeud).opérateur == ‘0’) Alors

Retourner ((\*pt\_noeud).valeur)

Sinon

valeur1 = évaluer((\*pt\_noeud).gauche)

valeur2 = évaluer((\*pt\_noeud).droit)

Si ((\*pt\_noeud).opérateur = ‘+’) Alors

Retourner (valeur1 + valeur2)

FinSi

Si ((\*pt\_noeud).opérateur = ‘-’) Alors

Retourner (valeur1 - valeur2)

FinSi

Si ((\*pt\_noeud).opérateur = ‘\*’) Alors

Retourner (valeur1 \* valeur2)

FinSi

Si ((\*pt\_noeud).opérateur = ‘/’) Alors

Retourner (valeur1 / valeur2)

FinSi

FinSi

Fin