DÉRIVÉES DE POLYNÔME

1.Méthode de Horner :

Référez-vous au cours pour plus de détails sur la méthode d'Hörner.

Il est possible de calculer la valeur d'un polynôme et de sa dérivée à l'aide de la méthode suivante dite méthode de Horner.

Coefs de $P_n(x)$	Calcul de $P_n(\alpha)$	Calcul de $P'_n(\alpha)$
a_0	$b_0 = a_0$	$c_0 = b_0$
a_1	$b_1 = b_0\alpha + a_1$	$c_1 = c_0 * \alpha + b_1$
$\vdots$	$\vdots$	$\vdots$
a_i	$b_i = b_{i-1}\alpha + a_i$	$c_i = c_{i-1} * \alpha + b_i$
$\vdots$	$\vdots$	$\vdots$
a_{n-1}	$b_{n-1} = b_{n-2}\alpha + a_{n-1}$	$c_{n-1} = c_{n-2} * \alpha + b_{n-1} = \frac{P'_n(\alpha)}{1!}$
a_n	$b_n = b_{n-1}\alpha + a_n = P_n(\alpha)$	

Pour calculer la dérivée en un point α donné, il suffit d'enchaîner deux schémas de Horner successifs.

2.Travail à effectuer :

Ecrire un programme qui :

- ➡ Effectue la saisie des coefficients d'un polynôme de degré inférieur ou égal à 9 et de son degré.
  La saisie sera recommencée tant que l'utilisateur n'aura pas saisi un degré compris dans l'intervalle [1 ; 9].
- ➡ Saisisse la valeur d'un point de départ **`x0`** ainsi qu'un pas **`h`** pour calculer la valeur du polynôme et de sa dérivée sur 26 points régulièrement espacés depuis le point de départ **`x0`**.
- ➡ Cherche dans les valeurs calculées une racine du polynôme et affiche la valeur de la dérivée en ce point. Dans le cas où il n'y aurait pas de racine le programme affichera le message : "pas de racine trouvée".

Pour ce faire, vous écrirez les fonctions et définitions de type suivantes. Les commentaires C expliquent le rôle de chaque variable.

```
 typedef float vect[10];
 typedef float pts[26];

void Horner( float *pa , int pn , float pal, float *pP, float *pb);
/* pa pointe vers le vecteur des coefs de P_n(x)
 * pn est le degre de P_n(x)
 * pal le point de calcul
 * pP pointe vers une variable qui contiendra P_n(al)
 * pb pointe vers un vecteur qui contiendra les b_i
 */

void saisie(float *pa , int *pn);
/* Saisie des coefs d'un polynome P_n(x)
 * pa pointe vers le vecteur qui contiendra les coefs de P_n(x)
 * pn pointe vers une variable qui contiendra le degre P_n(x)
 * La saisie sera refusée tant que n est >9
 */
```

3.Plan de travail

- ★ Calcul des valeurs prises par le polynôme :
  - ✦ Ecrivez la fonction **`saisie`**. Testez-la en affichant dans la fonction **`main`** les coefficients d'un polynôme saisi. Vérifiez que vous ne pouvez entrer un degré > 9, ni < 1.
  - ✦ Ecrivez la fonction **`Horner`**. Testez-la avec le polynôme suivant en α = 2 : $P_n(x) = x^2 + x - 3$

Vous devez trouver :

```
Coefs de P2(x) calcul de P2(2)
     1             b0 = 1
     1             b1 = 1*2+1 = 3
    -3             b2 = 3*2 -3 = 3 = P2(2)
```

- ✦ Ecrivez le code nécessaire pour réaliser la saisie de **`x0`** le point de départ et de **`h`** le pas. Ces saisies se feront dans la fonction **`main`**.
- ✦ À partir de **`x0`** et en utilisant le pas **`h`**, calculez 26 points équidistants. À l'aide de la fonction **`Horner`**, calculez les valeurs prises par le polynôme saisi avec la fonction **`saisie`** et calculez la valeur de la dérivée en ces points. Vous afficherez le résultat des calculs sous la forme de colonnes contenant les valeurs correspondant aux intitulés suivants :

```
n° du point : xi   Pn(xi)    P'n(xi)
```

Exemple avec $P_3(x) = x^3 + x^2 - 10x + 8$, point de départ $x_0 = -1$ pas : h = 0,1

```
0 :   -1.000     18.000          -9.000
1 :   -0.900     17.081          -9.370
2 :   -0.800     16.128          -9.680
3 :   -0.700     15.147          -9.930
…
```

★ Dans la fonction **`main`** écrivez le code nécessaire pour trouver les points dont la valeur absolue[1] de l'ordonnée est inférieure à 10^{-5}. Affichez alors le message suivant :

"Racine au point <indice du point> soit x= <abscisse correspondante> et P'(x) = <valeur de la dérivée>"

Exemple d'exécution du programme avec le même polynôme que ci-dessus :

```
Saisie des coefs du polynome :

Entrez le degre de P_n(x) : 3
Entrez le coef de x^3 : 1
Entrez le coef de x^2 : 1
Entrez le coef de x^1 : -10
Entrez le coef de x^0 : 8

Entrez le point de depart :-1
Entrez le pas :0.1

 0 :   -1.000     18.000          -9.000
 1 :   -0.900     17.081          -9.370
 2 :   -0.800     16.128          -9.680
 3 :   -0.700     15.147          -9.930
 4 :   -0.600     14.144         -10.120
 5 :   -0.500     13.125         -10.250
 6 :   -0.400     12.096         -10.320
 7 :   -0.300     11.063         -10.330
 8 :   -0.200     10.032         -10.280
 9 :   -0.100      9.009         -10.170
10 :    0.000      8.000         -10.000
11 :    0.100      7.011          -9.770
12 :    0.200      6.048          -9.480
13 :    0.300      5.117          -9.130
14 :    0.400      4.224          -8.720
15 :    0.500      3.375          -8.250
16 :    0.600      2.576          -7.720
17 :    0.700      1.833          -7.130
18 :    0.800      1.152          -6.480
19 :    0.900      0.539          -5.770
20 :    1.000      0.000          -5.000
21 :    1.100     -0.459          -4.170
22 :    1.200     -0.832          -3.280
23 :    1.300     -1.113          -2.330
24 :    1.400     -1.296          -1.320
25 :    1.500     -1.375          -0.250
Racine au point 20 soit x=1.000000
et P'(x) = -5.000000
```

[1] La fonction pour calculer la valeur absolue d'un **`float`** est **`fabs`**. Son prototype est dans **`math.h`**