ΕΡΓΑΣΤΗΡΙΑΚΗ ΑΣΚΗΣΗ 3.1

Πρόβλημα

- Δίνεται το Πολυώνυμο :

  $P(x) = p_0 x^n + p_1 x^{n-1} + ... + p_{n-1} x + p_n$

- Να γίνει πρόγραμμα C το οποίο να υπολογίζει τους συντελεστές του πηλίκου και το Υπόλοιπο της Διαίρεσης του Πολυωνύμου P(x) δια του Μονώνυμου :

  $x - \xi$

  με τη βοήθεια του σχήματος του Horner.

Αλγόριθμος main()

1. **Διαβάζουμε** τους συντελεστές $p_0, p_1, ..., p_n$ του Πολυωνύμου `P(x)` στον **πίνακα** `p`.
2. **Εμφανίζουμε** τους συντελεστές του Πολυωνύμου $p_0, p_1, ..., p_n$.
3. **Διαβάζουμε** το ξ.
4. **Υπολογίζουμε** τις τιμές του πηλίκου `Q(x)` στον πίνακα `q`, σύμφωνα με το σχήμα του Horner :

   a) $q_0 = p_0$
   b) **Για** τους συντελεστές $q_i, i = 1,2,...,n$:
      $q_i = p_i + q_{i-1} * \xi$

5. **Εμφανίζουμε** τις τιμές των συντελεστών $q_0, q_1, ..., q_{n-1}$
6. **Εμφανίζουμε** την τιμή του υπολοίπου της διαίρεσης (συντελεστής q_n).

3.1 Αριθμητικές Σταθερές

- Η δήλωση του πλήθους των στοιχείων των πινάκων `p` και `q` στο προηγούμενο πρόγραμμα μπορεί να γίνει με την εντολή :

```
#define n 3
```

Παρατηρήσεις

- Οι Σταθερές μπορούν να δηλωθούν στη αρχή του προγράμματος πριν τη συνάρτηση `main()` με την εντολή **#define** και προηγούνται των δηλώσεων των μεταβλητών ή με την εντολή **const** στην αρχή ή στο τμήμα δηλώσεων μεταβλητών. Π.χ.

```
const int n = 3;
```

- Οι Αριθμητικές Σταθερές στη C μπορεί να είναι Ακέραιες ή Πραγματικές και δεν αλλάζουν τιμή κατά τη διάρκεια του προγράμματος.

3.2 Δήλωση Μονοδιάστατων Πινάκων

- Η δήλωση γενικά ενός πίνακα `pin` `3` θέσεων για πραγματικούς αριθμούς γίνεται με την εντολή :

```
float pin[3];
```

με στοιχεία `pin[0], p[1], …, p[2].`

Παρατηρήσεις

- Η δήλωση του πλήθους των στοιχείων του πίνακα μπορεί να γίνει με τη δήλωση σταθεράς. Έτσι, θα είχαμε το ίδιο αποτέλεσμα με τις εντολές :

```
#define n 3
float pin[n];
```

3.3 Διάβασμα Στοιχείων ενός Μονοδιάστατου Πίνακα

- Με την ομάδα εντολών :

```
for ( i = 0; i<=n; i++ )
{
printf("Dose timh gia to p[%d ] : ",i);
scanf ("%f",&p[i]);
}
```

διαβάζουμε `n + 1 = 4` πραγματικούς αριθμούς απ' το πληκτρολόγιο χωρίς `format` **(έναν αριθμό σε κάθε γραμμή)** και τους αποθηκεύουμε στον πίνακα των συντελεστών `p`.

Παρατήρηση

- Αν θέλαμε να χρησιμοποιήσουμε το πρόγραμμα για συγκεκριμένο πολυώνυμο, π.χ. το $p(x) = x^3 - 4x$, θα μπορούσαμε να δώσουμε τις τιμές των στοιχείων του πίνακα `p` στην εντολή δήλωσης του πίνακα :

```
float ksi,q[n+1];
float p[n+1] = {1,0,-4,0};
```

3.4 Εμφάνιση n Στοιχείων Μονοδιάστατου Πίνακα

- Με την ομάδα εντολών :

```
for ( i = 0; i<=n; i++ )
{
printf("p[%d] = %f\n",i, p[i]);
}
```

εμφανίζουμε στην οθόνη `n + 1 = 4` πραγματικούς αριθμούς (τα στοιχεία του πίνακα των συντελεστών `p` και τον αντίστοιχο δείκτη) χωρίς `format`.

ΕΡΓΑΣΤΗΡΙΑΚΗ ΑΣΚΗΣΗ 3.2

- Να τροποποιηθεί η Άσκηση 3.1, ώστε ο Βαθμός του Πολυωνύμου να δίνεται απ' το πληκτρολόγιο.

Αλγόριθμος main()

1. **Διαβάζουμε** το Βαθμό του Πολυωνύμου **n**.

2. **Διαβάζουμε** τους συντελεστές $p_0, p_1, \ldots, p_n$ του Πολυωνύμου `P(x)` στον **πίνακα** `p`.

3. **Εμφανίζουμε** τους συντελεστές του Πολυωνύμου $p_0, p_1, \ldots, p_n$.

4. **Διαβάζουμε** το ξ.

5. **Υπολογίζουμε** τις τιμές του πηλίκου `Q(x)` στον πίνακα `q`, σύμφωνα με το σχήμα του `Horner`:

   a) $q_0 = p_0$
   b) **Για** τους συντελεστές $q_i, i = 1,2,\ldots,n$:
      $q_i = p_i + q_{i-1} * \xi$

6. **Εμφανίζουμε** τις τιμές των συντελεστών $q_0, q_1, \ldots, q_{n-1}$

7. **Εμφανίζουμε** την τιμή του υπολοίπου της διαίρεσης (συντελεστής $q_n = P(\xi)$).

3.5 Πέρασμα Πινάκων σαν Παραμέτρων σε functions

➢ Αν θέλαμε να χρησιμοποιήσουμε μια `function` που θα καλείται από το πρόγραμμα `main()` να υλοποιήσει το Βήμα 5 θα έπρεπε να περάσουμε τον **πίνακα** `p` σαν **παράμετρο**. Για το πέρασμα Πινάκων σαν παραμέτρων ισχύουν τα παρακάτω :

♦ Οι Πίνακες - Παράμετροι θα πρέπει να δηλωθούν στο κυρίως Πρόγραμμα.

♦ Ο Πίνακας περνάει σαν παράμετρος με το **όνομά** του και αγκύλες.

♦ Ο Πίνακας περνάει **με αναφορά** (by reference), οπότε, αν αλλάξει το περιεχόμενό του επιστρέφει τις νέες τιμές.

ΕΡΓΑΣΤΗΡΙΑΚΗ ΑΣΚΗΣΗ 3.3

- Να τροποποιηθεί η Άσκηση 3.2, ώστε ο Υπολογισμός του Υπολοίπου q_n να γίνεται με τη χρήση της συνάρτησης – `function Horner1()`, η κλήση της οποίας θα επιστρέφει το υπόλοιπο της διαίρεσης $P(x)/(x-\xi)$ και στην οποία θα εμφανίζουμε τους συντελεστές του πηλίκου **.**

Αλγόριθμος main()

1. **Διαβάζουμε** το Βαθμό του Πολυωνύμου **`n`**.
2. **Διαβάζουμε** τους συντελεστές $p_0, p_1, \ldots, p_n$ του Πολυωνύμου `P(x)` στον **πίνακα** `p`.
3. **Εμφανίζουμε** τους συντελεστές του Πολυωνύμου $p_0, p_1, \ldots, p_n$.
4. **Διαβάζουμε** το ξ.
5. **Εμφανίζουμε** την τιμή του υπολοίπου της διαίρεσης $P(x)/(x-\xi)$ με την κλήση της συνάρτησης **`Horner1(n,p,ksi)`** **.**

Αλγόριθμος function `Horner1()`

1) **Υπολογίζουμε** τις τιμές του πηλίκου `Q(x)` στον πίνακα `q`, σύμφωνα με το σχήμα του `Horner` :

   a) $q_0 = p_0$
   b) **Για** τους συντελεστές $q_i, i = 1,2,\ldots,n$:
      $q_i = p_i + q_{i-1} * \xi$

2) **Εμφανίζουμε** τους συντελεστές του πηλίκου $q_0, q_1, \ldots, q_{n-1}$
3) **Επιστρέφουμε το Υπόλοιπο** $q_n = P(\xi)$ **.**

ΕΡΓΑΣΤΗΡΙΑΚΗ ΑΣΚΗΣΗ 3.4

- Να τροποποιηθεί η Άσκηση 3.3, ώστε ο Υπολογισμός του Υπολοίπου q_n να γίνεται με τη χρήση της συνάρτησης – `function Horner2()`, η κλήση της οποίας θα επιστρέφει **όλες τις τιμές του πίνακα q**, ο οποίος πρέπει να περάσει σαν **παράμετρος**, έτσι ώστε να εμφανίσουμε τους συντελεστές του πηλίκου στη `main()`.

Αλγόριθμος main()

1) **Διαβάζουμε** το Βαθμό του Πολυωνύμου **n**.

2) **Διαβάζουμε** τους συντελεστές $p_0, p_1, \ldots, p_n$ του Πολυωνύμου `P(x)` στον **πίνακα** `p`.

3) **Εμφανίζουμε** τους συντελεστές του Πολυωνύμου $p_0, p_1, \ldots, p_n$.

4) **Διαβάζουμε** το ξ.

5) **Εμφανίζουμε** την τιμή του υπολοίπου της διαίρεσης $P(x)/(x-\xi)$ με την κλήση της συνάρτησης **`Horner2(n,p,q,ksi)`**.

6) **Εμφανίζουμε** τους συντελεστές του πηλίκου $q_0, q_1, \ldots, q_{n-1}$

Αλγόριθμος function `Horner2()`

1) **Υπολογίζουμε** τις τιμές του πηλίκου `Q(x)` στον πίνακα `q`, σύμφωνα με το σχήμα του `Horner` :

   a) $q_0 = p_0$
   b) **Για** τους συντελεστές $q_i, i = 1,2,\ldots,n$:
   $$q_i = p_i + q_{i-1} * \xi$$

2) **Επιστρέφουμε το Υπόλοιπο** $q_n = P(\xi)$.

ΕΡΓΑΣΤΗΡΙΑΚΗ ΑΣΚΗΣΗ 3.5

- Να τροποποιηθεί η Άσκηση 3.4, ώστε ο Υπολογισμός του Υπολοίπου q_n να γίνεται με τη χρήση της συνάρτησης – `function Horner2`, η κλήση της οποίας θα επιστρέφει **όλες τις τιμές του πίνακα** `q`, ο οποίος πρέπει να περάσει σαν παράμετρος, έτσι ώστε να ξανακαλέσουμε τη συνάρτηση για τον υπολογισμό και της **πρώτης παραγώγου**.

Αλγόριθμος main()

1. **Διαβάζουμε** το Βαθμό του Πολυωνύμου **`n`**.
2. **Διαβάζουμε** τους συντελεστές $p_0, p_1, \ldots, p_n$ του Πολυωνύμου `P(x)` στον **πίνακα** `p`.
3. **Εμφανίζουμε** τους συντελεστές του Πολυωνύμου $p_0, p_1, \ldots, p_n$.
4. **Διαβάζουμε** το ξ.
5. **Εμφανίζουμε** την τιμή του υπολοίπου της διαίρεσης $P(x)/(x-\xi)$ με την κλήση της συνάρτησης **`Horner2(n,p,q,ksi)`**
6. **Εμφανίζουμε** τους συντελεστές του πηλίκου $q_0, q_1, \ldots, q_{n-1}$
7. **Εμφανίζουμε** την τιμή του υπολοίπου της διαίρεσης $Q(x)/(x-\xi)$ με την κλήση της συνάρτησης **`Horner2(n-1,q,r,ksi)`**
8. **Εμφανίζουμε** τους συντελεστές του νέου πηλίκου $r_0, r_1, \ldots, r_{n-2}$.

Αλγόριθμος function `Horner2(n,p,q,ksi)`

1) **Υπολογίζουμε** τις τιμές του πηλίκου `Q(x)` στον πίνακα `q`, σύμφωνα με το σχήμα του `Horner` :

   a) $q_0 = p_0$
   b) **Για** τους συντελεστές $q_i, i = 1,2,\ldots,n$:
      $q_i = p_i + q_{i-1} * \xi$

2) **Επιστρέφουμε το Υπόλοιπο (** q_n **)**

ΕΡΓΑΣΤΗΡΙΑΚΗ ΑΣΚΗΣΗ 3.6

Πρόβλημα

- Να γίνει πρόγραμμα C, το οποίο να προσομοιώνει τη συνάρτηση `sqrt(num)` για τον υπολογισμό της τετραγωνικής ρίζας ενός αριθμού num, λύνοντας την εξίσωση $x = \sqrt{num} \Rightarrow x^2 = num \Rightarrow f(x, num) = x^2 - num = 0$. Η εξίσωση θα δηλώνεται με την εντολή `#define`, ενώ ο υπολογισμός της τετραγωνικής ρίζας θα γίνεται με τη βοήθεια της μεθόδου `Newton-Raphson`.

Αλγόριθμος main()

1. **Διαβάζουμε** τον αριθμό **`num`**.
2. **Δίνουμε** αρχική τιμή στο `x` και στο `oldx`.
3. **Εμφανίζουμε** την τιμή του `num`, του `x` και του **`sqrt(num)`**
4. **Για Όσο** $\left(|x - oldx| > 10^{-15}\right)$

    Αποθηκεύουμε το x στο oldx

    Υπολογίζουμε το νέο x σύμφωνα με τη μέθοδο `Newton-Raphson` $x = x - \frac{f(x)}{f^{'}(x)}$

    Εμφανίζουμε την τιμή του `num`, του `x` και του **`sqrt(num)`**

ΕΡΓΑΣΤΗΡΙΑΚΗ ΑΣΚΗΣΗ 3.7

- Να τροποποιηθεί η Άσκηση 3.6, ώστε ο υπολογισμός της τετραγωνικής ρίζας ενός αριθμού `num` να γίνεται με την κλήση της συνάρτησης `mysqrt(num)`.

Αλγόριθμος main()

1. **Διαβάζουμε** τον αριθμό **`num`**.
2. **Εμφανίζουμε** την τιμή του `num`, του **`mysqrt(num)`** και του **`sqrt(num)`**

Αλγόριθμος function `mysqrt(num)`

1. **Δίνουμε** αρχική τιμή στο `x` και στο `oldx`.
2. **Για Όσο** $\left(|x - oldx| > 10^{-15}\right)$

    Αποθηκεύουμε το x στο oldx

    Υπολογίζουμε το νέο x σύμφωνα με τη μέθοδο Newton-Raphson
    $$x = x - \frac{f(x)}{f'(x)}$$

ΕΡΓΑΣΤΗΡΙΑΚΗ ΑΣΚΗΣΗ 3.8

- Να γίνει πρόγραμμα C, το οποίο να βρίσκει **όλες τις ρίζες** μιας εξίσωσης που περιέχονται στο διάστημα `[a, b]` με τη βοήθεια της μεθόδου `Newton-Raphson`. Το διάστημα `[a, b]` θα χωρίζεται με το βήμα `h` στα υπο-διαστήματα $[a,a+h]$, $[a+h,a+2h]$,…., $[b\text{-}h,b]$, στο καθένα απ' τα οποία θα ελέγχεται αν υπάρχει ρίζα. Σ' αυτή την περίπτωση, θα χρησιμοποιείται ο αλγόριθμος 3.6 για την εύρεσή της. Η εξίσωση θα δηλώνεται με την εντολή `#define`.

Αλγόριθμος main()

1. **Διαβάζουμε** τα όρια του διαστήματος των ριζών **`a, b`** και το βήμα **`h`**.
2. **Για** το κάθε υποδιάστημα $[a,a+h],[a+h,a+2h],...,[b-h,b]$

   Αν υπάρχει ρίζα στο υποδιάστημα **τότε**

   Δίνουμε αρχική τιμή στο `x` και στο `oldx` τα άκρα του υποδιαστήματος

   Εμφανίζουμε την τιμή του `x` και του **`f(x)`**

   Για Όσο $\left(|x-oldx|>10^{-15}\right)$

   Αποθηκεύουμε το x στο oldx

   Υπολογίζουμε το νέο x σύμφωνα με τη μέθοδο `Newton-Raphson` $x=x-\dfrac{f(x)}{f'(x)}$

   Εμφανίζουμε την τιμή του `x` και του **`f(x)`**

ΕΡΓΑΣΤΗΡΙΑΚΗ ΑΣΚΗΣΗ 3.9

Πρόβλημα

- Δίνεται το Πολυώνυμο $P(x) = p_0x^n + p_1x^{n-1} + ... + p_{n-1}x + p_n$. Να γίνει πρόγραμμα C, το οποίο να υπολογίζει **όλες** τις ρίζες του Πολυωνύμου που περιέχονται στο διάστημα `[a, b]` με τη βοήθεια της μεθόδου των **Newton-Raphson.** Το διάστημα `[a, b]` θα χωρίζεται με το βήμα `h` στα υπο-διαστήματα $[a,a+h]$, $[a+h,a+2h]$, …. , $[b-h,b]$, στο καθένα απ' τα οποία θα ελέγχεται αν υπάρχει ρίζα. Σ' αυτή την περίπτωση, θα χρησιμοποιείται ο αλγόριθμος 3.6 για την εύρεσή της. Για τον υπολογισμό της Τιμής του Πολυωνύμου και της τιμής της Παραγώγου του Πολυωνύμου (οπουδήποτε στον αλγόριθμο εμφανίζεται το **`P(x)`** ή το **`P' (x)`**)θα χρησιμοποιηθεί η συνάρτηση Horner της Εργαστηριακής Άσκησης 3.4.

Αλγόριθμος main()

1. **Διαβάζουμε** το Βαθμό του Πολυωνύμου **`n`**.
2. **Διαβάζουμε** τους συντελεστές $p_0,p_1,...,p_n$ του Πολυωνύμου `P(x)` στον **πίνακα** `p`.
3. **Εμφανίζουμε** τους συντελεστές του Πολυωνύμου $p_0,p_1,...,p_n$.
4. **Διαβάζουμε** τα όρια του διαστήματος των ριζών **`a, b`** και το βήμα **`h`**.
5. **Για** το κάθε υποδιάστημα $[a,a+h],[a+h,a+2h],...,[b-h,b]$

   Αν υπάρχει ρίζα στο υποδιάστημα **τότε**

   Δίνουμε αρχική τιμή στο `x` και στο `oldx` τα άκρα του υποδιαστήματος

   Εμφανίζουμε την τιμή του `x` και του **`P(x)`**

   Για Όσο $\left(|x - oldx| > 10^{-15}\right)$

   Αποθηκεύουμε το x στο oldx

   Υπολογίζουμε το νέο x σύμφωνα με τη μέθοδο `Newton-Raphson` $x = x - \dfrac{P(x)}{P'(x)}$

   Εμφανίζουμε την τιμή του `x` και του **`P(x)`**