Université : A.Mira – Bejaia
Faculté : Sciences Exactes
Département : Informatique

Année Universitaire : 2022-2023
Niveau : Première Année Ingénieur

EXAMEN FINAL – ALSD1 (SEMESTRE 1)

Exercice 01 (8 pts)

Soit l'algorithme suivant :

Algorithme

```
Algorithme Exo1;
  Variables
      n, i, f, d : entier;
      x, p, S : réel;
Début
    Lire(n, x) ;
    S ← 0;  p ← 1; f ← 1; d←0 ;
    Pour i←0 à (n-1) faire
      d ← d + f;
      S ← S + p / d;
      f ← f * (i+1);
      p ← p * x * x;
    Fin-Pour
    Écrire('Le Résultat S =  ', S);
Fin.
```

Questions :

1- Traduire l'algorithme en programme C

2- Dérouler l'algorithme pour n=4 et x = 2

3- Déduire l'expression du résultat calculé par l'algorithme en fonction des variables d'entrée.

4- Récrire l'algorithme en remplaçant la boucle ***Pour*** par la boucle ***Tant-que***

5- Modifier l'algorithme pour calculer le même résultat avec des signes alternatifs des termes :
S = Terme1 – Terme2 + Terme3 - .... ± TermeNS

1.5

1- Traduction en programme C

Programme C

```
#include <stdio.h>
int main()
{
    int n, i, f, d ;        0.25
    float x, p, S ;

    scanf("%d %f", &n, &x) ;  0.25
    S = 0;  p = 1; f = 1; d = 0 ;    0.25
    for (i=0 ; i<n ; i++)
    {
        d = d + f;
        S = S + p / d;             0.5
        f = f * (i+1);
        p = p * x * x;
    }
    printf("Le Résultat S = %f", S);  0.25
}
```

3.0

2- Dérouler l'algorithme pour n=4 et x=2

Instructions	Variables							Affichage
	n	x	i	d	f	p	s	
Lire(n, x) ;	4	2	/	/	/	/	/ 0.25	
S ← 0; p ← 1; f ← 1; d←0 ;	''	''	/	0	1	1	0 0.25	
Pour i ← 0 d ← d + f = 0 + 1 = 1 S ← S + p / d = 0 + 1 / 1 = 1 f ← f * (i+1) = 1*(0+1) = 1 p ← p * x * x = 1*x^2 = 2^2 = 4	''	''	0	0 1	1 1	1 x^2= 4	0 **1** 0.5	
Pour i ← 1 d ← d + f = 1 + 1 = 2 S ← S + p / d = 1 + 4 / 2 = 3 f ← f * (i+1) = 1*(1+1)=1*2=2 p ← p * x * x = 4*x^2 = 2^4 = 16	''	''	1	1 2	1 2	1 x^4=16	1 1 + 4/2 = **3** 0.5	
Pour i ← 2 d ← d + f = 2 + 2 = 4 S ← S + p / d = 3 + 16 / 4 = 7 f ← f * (i+1) = 2*(2+1)=2*3=6 p ← p * x * x = 16*x^2 = 2^6 = 64	''	''	2	2 4	2 6	16 x^6=64	3 3 + 16/4 = **7** 0.5	
Pour i ← 3 d ← d + f = 4 + 6 = 10 S ← S + p / d = 7 + 64 / 10 = 13.4 f ← f * (i+1) = 6*(3+1)=6*4=24 p ← p * x * x = 64*x^2 = 2^8 = 256	''	''	3	4 10	6 24	64 256	7 7+64/10 = **13.4** 0.5	
Écrire('Le Résultat S = ', S);	''	''	''	''	''	''	**13.4** 0.5	Le résultat = **13.4**

Donc, l'algorithme affiche : ***Le résultat S = 13.4***

1.0

3- Déduire l'expression de S

L'expression du Résultat (variable de sortie) S en fonction des entrées (c'est à dire *n* et x). D'après le déroulement, on comprend le rôle de chaque variable :

d : représente la somme des f, on peut écrire d = Σf

f : représente le factoriel de i (au début de l'itération f = i!) et à la fin de l'itération f= (i+1)!

S : représente la somme de P / d, donc, on peut écrire S = Σ (p / d) = Σ(p / Σf)

p : représente la puissance de x, au début de l'iération, $p=x^{2i}$ et à la fin de l'itération, $p=x^{2(i+1)}$

Si, on regroupe tous ces éléments, on trouve :

$$S=\frac{x^0}{0!}+\frac{x^2}{0!+1!}+\frac{x^4}{0!+1!+2!}+\ldots+\frac{x^{2*(n-1)}}{0!+1!+2!+\ldots.+(n-1)!}=\sum_{i=0}^{n-1}\frac{x^{2*i}}{0!+1!+\ldots+i!}=\sum_{i=0}^{n-1}\frac{x^{2*i}}{\sum_{j=0}^{i} j!}$$

L'une des trois expressions

1.25

4- Réécrire l'algorithme en utilisant la boucle Tant-que

Algorithme
Algorithme Exo1; Variables n, i, f, d : entier; x, p, S : réel; Début Lire(n, x) ; S ← 0; p ← 1; f ← 1; d←0 ; i ← 0; **0.25** Tant-que (i<= (n-1)) faire **0.5** d ← d + f; S ← S + p / d; f ← f * (i+1); p ← p * x * x; i ← i+1; **0.5** Fin-Tant-que; Écrire('Le Résultat S = ', S); Fin.

1.25

5- Modifier l'algorithme pour calculer la somme alternée

Algorithme
Algorithme Exo1; Variables **0.25** n, i, f, d, k : entier; x, p, S : réel; Début Lire(n, x) ; S ← 0; p ← 1; f ← 1; d←0 ; k ← 1; **0.25** Pour i←0 à (n-1) faire d ← d + f; S ← S + k * p / d; **0.25** f ← f * (i+1); p ← p * x * x; k ← -k; **0.5** Fin-Pour Écrire('Le Résultat S = ', S); Fin.

Exercice N°02 (8 pts)

Soit T un tableau de type entier et de taille N.
Écrire un algorithme qui permet de réaliser les opérations suivantes :

1) Lire le du tableau T de taille N (N ≥ 4).

2) Supprimer toutes les valeurs négatives du tableau T.

3) Trier les éléments du vecteur T avec ordre croissant

4) Fragmenter le tableau en deux tableau T1 et T2, sachant que T1 contient uniquement les entiers premiers ,T2 contient les autres éléments de T.

Algorithme

```
Algorithme Exo2;
  Variables
      T, T1, T2 : Tableau[1..100] d'entiers;
      N, i, j, imin, Z, Nbd : entier;
      N1, N2 : entier;
Début
   // Q1 – Lire le vecteur T   1
    Répéter
       Lire(n) ;
    Jusqu'à (n>=4) ;
    Pour i←1 à N faire
      Lire(T[i]);
    Fin-Pour;

    // Q2 – Supprimer toutes les valeurs négatives   2
    Pour i←1 à N faire
       Si (T[i] <= 0) alors
          Pour j← i à (N-1) faire
             T[j] ← T[j+1]; // Décalage à gauche ...
          Fin-Pour;
          N ← N – 1; // On a supprimé un élément ...
       Fin-Si;
    Fin-Pour ;

    // Q3 – Trier avec ordre croissant   2.25
    Pour i←1 à (N-1) faire
       imin ← i;
       Pour j← (i+1) à N faire
         Si (T[j] < T[imin]) alors
            imin ← j;
         Fin-si;
       Fin-pour;
       Si imin <> i alors
          Z ← T[imin] ; T[imin] ← T[i] ; T[i] ← Z;
       Fin-si;
    Fin-Pour;

    // Q4 – Fragmenter (diviser) le vecteur T en T1 et T2   2.25
    N1← 0 ;  N2←0;
    Pour i←1 à N faire
       nbd ← 0;
       Pour j←1 à T[i] faire
          Si T[i] mod j = 0 alors nbd ← nbd+1; fin-si;
       Fin-Pour;
       Si nbd =2 alors //T[i] est premier
            N1 ←  N1+1 ;   T1[N1] ← T[i];
       Sinon    //T[i] n'est pas premier
            N2 ←  N2+1 ;   T2[N2] ← T[i];
       Fin-Si;
    Fin-Pour;

    // Affichage de T1 et T2
    Pour i←1 à N1 faire  écrire(T1[i]) ; Fin-Pour;   0.25
    Pour i←1 à N2 faire  écrire(T2[i]) ; Fin-Pour;   0.25
Fin.
```

Exercice N°03 (4 pts)

Nous voulons calculer la valeur de N=(x! - (3y-x)!)*z! Sachant que x, y et z sont des entiers strictement positives (lus au clavier).

1- Écrire un algorithme dans lequel :

- On déclare les variables globales nécessaires
- On déclare un sous programme (fonction) *factoriel* qui calcul le factoriel d'un entier
- On écrit les instructions qui permettent de calculer N en appelons le sous programme *factoriel*.

2- Réécrire l'algorithme de la question 1 en remplaçant la fonction *factoriel* par une procédure *factoriel*.

2 Q1 – Écriture de l'algorithme

```
Algorithme Exo3_1;
  Variables
      x, y, z, N : entier;  0.25

  Fonction factoriel(n:entier) : entier;
    Variables  i, f : entier;
  Début
    f ← 1;
    Pour i←2 à n faire                      0.75
       f ← f*i;
    Fin-Pour;

    retourner f;
  Fin;

Début
    Répéter
       Lire(x) ;
    Jusqu'à x>0;
    Répéter
       Lire(y) ;
    Jusqu'à y>0;
    Répéter
       Lire(z) ;
    Jusqu'à x>0;

    N ← ( factoriel(x) – factoriel(3*y – x)) * factoriel(z);
                                0.75
    Écrire (N);
Fin.          0.25
```

2 Q2 – Fonction → Procédure

```
Algorithme Exo3_1;
  Variables
       x, y, z, N : entier;
       f1, f2, f3 : entier;0.25

  Procédure factoriel(n:entier ; E/S f : entier);
    Variables  i : entier;
  Début
    f ← 1;
    Pour i←2 à n faire
       f ← f*i;                                0.75
    Fin-Pour;
  Fin;

Début
    Répéter
       Lire(x) ;
    Jusqu'à x>0;
    Répéter
       Lire(y) ;
    Jusqu'à y>0;
    Répéter
       Lire(z) ;
    Jusqu'à x>0;

    factoriel(x, f1);
    factoriel(3*y – x, f2);   0.75
    factoriel(z, f3);

    N ← (f1  – f2) * f3; 0.25

    Écrire (N);
Fin.
```