Université des Sciences et de la Technologie Houari Boumediene

Conception et Complexité des Algorithme

Rapport de Travaux Pratiques N°1
Mesure du temps d'exécution d'un programme.

Quadrinôme
MOHAMMEDI Haroune
HOUACINE Naila Aziza

Professeur
Pr. AMANI Ferhat

14 octobre 2017

1 Partie I : Développement de l'algorithme et du programme pour le problème du calcul de la somme des n premiers nombres entiers naturels.

1 Développement de l'algorithme itératif qui permet de calculer la somme S des n premiers nombres entiers naturels.

L'écriture de ces algorithme sera accompagné de commentaire représentant le nombre de mot mémoire que prend chaque instruction puis celui de tous l'algorithme.

1.1 En utilisant la forme "pour ... faire" :

L'algorithme développé cidessous, nommé Algorithme_Somme_Iteratif_pour, utilise la forme de répétition : Pour ... faire ... fait.

```
 Algorithme_Somme_Iteratif_pour

VAR
i,N,S : entier;                 //3 mots mémoire

Debut
   ecrire("Donner N = ");       //1 mot mémoire
   lire(N);                     //1 mot mémoire

   S = 0;                       //1 mot mémoire

   pour(i=1 ; i <= N ; i++)     //4 mots mémoire
     faire
       S = S + i;               //2 mots mémoire
     fait;

   ecrire("La somme = ",S);     //1 mot mémoire

Fin.
```

Totale de mots mémoire = 13 MM

1.2 En utilisant la forme "tant que ... faire" :

L algorithme développé cidessous, nommé Algorithme_Somme_Iteratif_tant_que, utilise la forme de répétition : tant que ... faire ... fait.

```
Algorithme_Somme_Iteratif_tant_que

VAR
i,N,S : entier;                 //3 mots mémoire

Debut
   ecrire("Donner N = ");       //1 mot mémoire
   lire(N);                     //1 mot mémoire

   S = 0;                       //1 mot mémoire
   i = 1;                       //1 mot mémoire

   tant que(i <= N)             //1 mot mémoire
     faire
       S = S + i;               //2 mots mémoire
       i = i + 1;               //2 mots mémoire
     fait;
```

```
    ecrire("La somme = ",S);     //1 mot mémoire

Fin.
```

Totale de mots mémoire = 13 MM

1.3 En utilisant la forme "répéter ... jusqu'à" :

L algorithme développé cidessous, nommé Algorithme_Somme_Iteratif_répéter_jusqu_a, utilise la forme de répétition : répéter ... jusqu'à ...

```
 Algorithme_Somme_Iteratif_répéter_jusqu_a

VAR
i,N,S : entier;                 //3 mots mémoire

Debut
    ecrire("Donner N = ");      //1 mot mémoire
    lire(N);                    //1 mot mémoire

    S = 0;                      //1 mot mémoire
    i = 1;                      //1 mot mémoire

    répéter
        S = S + i;              //2 mots mémoire
        i = i + 1;              //2 mots mémoire
      jusqu a(i > N);           //1 mot mémoire

    ecrire("La somme = ",S);    //1 mot mémoire

Fin.
```

Totale de mots mémoire = 13 MM

2 Développement des programmes itératifs en langage C.

2.1 En utilisant la forme "FOR" :

```
#include<stdio.h>
#include<stdlib.h>

int main()
{
    long int i,N,S;

    printf("Donner N = ");
    scanf("%Ld",&N);

    S=0;

    for(i=1 ; i <= N ; i++)
    {
        S = S + i;
    }

    printf("La somme S = %Ld",S);
    return 0;
}
```

2.2 En utilisant la forme "WHILE" :

```
#include<stdio.h>
#include<stdlib.h>

int main()
{
    long int i,N,S;

    printf("Donner N = ");
    scanf("%Ld",&N);

    S=0; i=1;

    while(i <= N)
    {
        S = S + i;
        i = i + 1;
    }

    printf("La somme S = %Ld",S);
    return 0;
}
```

2.3 En utilisant la forme "DO ... WHILE" :

```
#include<stdio.h>
#include<stdlib.h>

int main()
{
    long int i,N,S;

    printf("Donner N = ");
    scanf("%Ld",&N);

    S=0; i=1;

    Do
    {
        S = S + i;
        i = i + 1;
    }while(i > N)

    printf("La somme S = %Ld",S);
    return 0;
}
```

3 Développement de l'algorithme récursif pour le problème.

```
Procedure_Somme_Récursif(N , S : entier)

 VAR

 Debut
```

```
    Si( N <= 0)                                          //1 mot mémoire
        Alors ecrire("La Somme S = ",S);                 //1 mot mémoire
        Sinon   S = S + N;                               //2 mots mémoire
                Algorithme_Somme_Récursif(N - 1,S); //2 mots mémoire
    FinSi;

 Fin.


 Algorithme_Somme_Récursif()

 VAR

 Debut
    N: entier;                                           //1 mot mémoire

    ecrire("Donner N = ");                               //1 mot mémoire
    lire(N);                                             //1 mot mémoire

    Procedure_Somme_Récursif(N , 0);                     //1 mot mémoire
 Fin.
```

Totale de mots mémoire = 13 MM

4 Développement du programme récursif en langage C.

```c
#include<stdio.h>
#include<stdlib.h>

void SommeRecursive(long int N,long int S)
{
    if(N <= 0)
    {
        printf("La somme S = %Ld \n",S);
    }else
    {
        S = S + N;
        SommeRecursive(N - 1,S);
    }
}


int main()
{
    long int N;

    printf("Donner N = ");
    scanf("%Ld",&N);

    SommeRecursive(N,0);

    return 0;
}
```

2 Partie II : Mesure du temps d'exécution.

Pour la mesure du temps en langage C nous aurons besoin des fonctions de la gestion du temps que l'on retrouve dans la bibliothèque time.h.

Donc nous devant d'abord inclure la directive #include ¡time.h¿ Puis définir deux(2) variables notées début et fin de type clock_t.

A l'aide de la fonction "clock()" : Récupéré le Temps avant l'exécution du programme dans début et récupéré le Temps après la fin de l'exécution du programme dans fin. Calculer la différence entre début et fin dans un variable nommé : Delta de type double. Afin que le temps d'exécution Delta soit donné en secondes, nous divisons ce dernier par le paramètre CLOCKS_PER_SEC.

5 Mesurer des temps d'exécution

6 Représentation par un graph

7 Déduction du temps d'exécution moyen