

TP d'Algorithmique n°1

Table des matières

Exercice 1 : Fonction Récursive	2
Rappel de l'énoncé	2
Analyse du travail à faire	2
Algorithmes	2
Programmes	2
Jeux d'essai	3
Exercice 2 : Calcul Récursif	3
Rappel de l'énoncé	3
Analyse du travail à faire	3
Algorithmes	3
Programmes	4
Jeux d'essai	5
Exercice 3 : PGCD	6
Rappel de l'énoncé	6
Analyse du travail à faire	6
Algorithmes	7
Programmes	7
Jeux d'essai	8

Exercice 1 : Fonction Récursive

Rappel de l'énoncé

Nous devons écrire une fonction récursive qui calcule la somme des N premiers entiers.

Analyse du travail à faire

Nous devons d'abord déterminer la relation de récurrence afin de pouvoir appliquer la fonction récursivement, puis une condition d'arrêt pour éviter une boucle infinie.

Algorithmes

//Description : Fonction qui calcule la somme des n premiers entiers de façon récursive

//Condition d'arrêt : si le nombre = 1 alors la somme = 1

//pre-condition : Avoir un entier initialisé

// post-condition : Retourne un entier

Fonction sommeRecursive(n : Entier) Retourne Entier

Debut

Si n = 1 Alors Retourne 1

Sinon Retourne n+ sommeRecursive(n – 1)

Fin Si

Fin sommeRecursive

Programmes

```
int sommeRecursive(int chiffre) {  
    if (chiffre == 0) { //Condition d'arrêt  
        return 0;  
    }  
    else {  
        return (chiffre + sommeRecursive(chiffre - 1)); //Appel récursif de  
la fonction  
    }  
}
```

Jeux d'essai

Jeux d'essais où :

- le chiffre vaut 0
- le chiffre vaut 1
- le chiffre vaut 3
- le chiffre vaut 25

```
Entrer le chiffre :  
0  
Resultat : 0  
Entrer le chiffre :  
1  
Resultat : 1  
Entrer le chiffre :  
3  
Resultat : 6  
Entrer le chiffre :  
25  
Resultat : 325
```

Exercice 2 : Calcul Récursif

Rappel de l'énoncé

Nous devons calculer récursivement a^n avec n un entier positif ou nul, de telle sorte que $a^{n/2}$ et $a^{(n-1)/2}$ ne soit calculé qu'une seule fois.

Analyse du travail à faire

Nous devons avoir deux conditions d'arrêt, le cas où la puissance = 1 et le cas où elle est égale à 0, puis les cas où le nombre est pair ou impair.

Algorithmes

Fonction puissanceRecursive(chiffre : Entier, puissance : Entier) Retourne Entier
Début

Avec : resultat : Entier

Si puissance = 0 Alors // Condition d'arrêt
Retourne 1

Fin Si

Si puissance = 1 Alors // Cas de la puissance = 1
Retourne chiffre

Fin Si

Si chiffre mod 2 = 0 Alors // Cas puissance paire
 resultat -> puissanceRecursive(chiffre, puissance/2) //Appel récursif
de la fonction

Retourne resultat * resultat

Sinon // Cas puissance impaire

 resultat -> puissanceRecursive(chiffre, ((puissance - 1)/2));

 //Appel récursif de la fonction

Retourne chiffre * resultat * resultat

Fin Si

Fin puissanceRecursive

Programmes

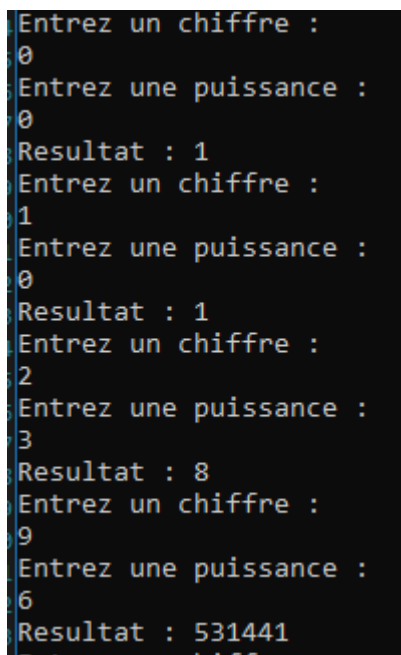
```
int puissanceRecursive(int chiffre, int puissance) {  
  
    int resultat;  
  
    if (puissance == 0) {                                // Condition d'arrêt  
        return 1;  
    }  
  
    if(puissance == 1) {                                  // Cas de la puissance = 1  
        return chiffre;  
    }  
  
    if((puissance%2) == 0){                                // Cas puissance paire  
        resultat = puissanceRecursive(chiffre, puissance/2);    // Appel récursif de la  
fonction  
        return (resultat * resultat);  
    }  
    T1_KURTH_PICHON
```

```
    }else {                                     // Cas puissance impaire
        resultat = puissanceRecursive(chiffre, ((puissance - 1)/2));    // Appel récursif de la
fonction
        return (chiffre * resultat * resultat);
    }
}
```

Jeux d'essai

Jeux d'essais où :

- la puissance et le chiffre valent 0
- la puissance vaut 0 et le chiffre 1
- plus tests communs



```
Entrez un chiffre :
0
Entrez une puissance :
0
Resultat : 1
Entrez un chiffre :
1
Entrez une puissance :
0
Resultat : 1
Entrez un chiffre :
2
Entrez une puissance :
3
Resultat : 8
Entrez un chiffre :
9
Entrez une puissance :
6
Resultat : 531441
```

Exercice 3 : PGCD

Rappel de l'énoncé

Nous devons écrire un programme calculant le PGCD de deux nombres en utilisant la récursivité

Analyse du travail à faire

Pendant cet exercice nous devons veiller au cas où le nombre A est égal au nombre B, où $A > B$ et où $A < B$.

Algorithmes

Fonction PGCD(entier1 : Entier , entier2 : Entier) Retourne Entier

Début

Avec :

Si entier1 = entier2 Alors // Cas A = B

Retourne entier1

Fin Si

Si entier1 > entier2 Alors // cas A > B

Retourne PGCD(entier1 - entier2, entier2)

// Appel récursif de la fonction

Sinon

Retourne PGCD(entier1, entier2 - entier1) // Appel récursif de la

fonction

Fin Si

Fin PGCD

Programmes

```
int PGCD(int entier1, int entier2) {
    if (entier1 == entier2) { // Cas A = B
        return entier1;
    }

    if (entier1 > entier2) { // cas A > B
        return PGCD(entier1 - entier2, entier2); // Appel récursif de la
fonction
    }
    else {
        return PGCD(entier1, entier2 - entier1); // Appel récursif de la
fonction
    }
}
```

Jeux d'essai

Jeux d'essais pour les cas où

- les deux chiffres sont égaux
- le premier est inférieur au deuxième
- le premier est supérieur au deuxième

```
Entrez un chiffre :  
2  
Entrez un chiffre :  
2  
Resultat : 2  
Entrez un chiffre :  
25  
Entrez un chiffre :  
39  
Resultat : 1  
Entrez un chiffre :  
39  
Entrez un chiffre :  
99  
Resultat : 3  
Entrez un chiffre :  
25  
Entrez un chiffre :  
5  
Resultat : 5  
Entrez un chiffre :  
5  
Entrez un chiffre :  
25  
Resultat : 5
```