– Algorithmique – TD N°4 –Procédures & Fonctions & Récursivité

Exercice 1 : Ordre croissant

Q1) Ecrire une procédure **Trier2** qui prend deux (2) entiers **A** et **B**, ensuite elle les permute, si nécessaire, pour que l'état de sortie soit **A ≤ B**. C'est-à-dire que **A** et **B** seront dans l'ordre croissant.

Q2) En s'inspirant du tri à bulles, écrire une procédure **Trier3** sur trois (3) entiers **(A, B et C)** qui appelle**Trier2** et permet d'avoir les trois variables **A, B et C** dans l'ordre croissant **(A ≤ B ≤ C)**.

Q3) Ecrire l'algorithme principal qui permet de lire 3 entiers **(X, Y et Z)** et de les afficher dans l'ordre croissant.

Exercice 2 : Carré Parfait

Q1) Ecrire une procédure **CParfait** qui permet de vérifier si un entier positif **N** est un **carré parfait**.

Cette procédure donne deux résultats : un booléen qui est égal à **vrai** si et seulement si **N** est un carré parfait et un entier correspondant à la partie entière de la racine carré de **N**.

Un entier **N** est **carré parfait** s'il est le carré d'un entier **k**, c'est-à-dire **N = k²**. Par exemple, les entiers 0, 1, 4, 9, 16 et 25 sont des carrés parfaits. Pour chercher la valeur de **k**, on calcule la somme des **k** premiers nombres impairs jusqu'à ce que cette somme soit supérieure ou égale à **N**.

-Si **N=16** alors S=1+3+5+7= la sommedes4premiersnombresimpairs ; **S=N** ➔ 16est carré parfait et $\sqrt{16}$ **= 4.**

-Si **N = 18** alors S = 1+3+5+7+9 = la somme des 5 premiers nombres impairs ; **S > N** ➔18 n'est pas carré parfait et **la partie entière de** $\sqrt{18}$ **est égale à 4.**

Note : L'utilisation des fonctions SQR (le carré) et SQRT (la racine carrée) n'est pas autorisée.

Q2) En utilisant la procédure **CParfait**, écrire l'algorithme principal qui permet de :

①**-**Lire un entier positif **N**, ②**-** Vérifier si **N** est un carré parfait et afficher sa racine carrée, sinon afficher la partie entière de sa racine carrée.

Exercice 3 : Nombres Proniques

Un entier positif **N** est un **nombre pronique** (presque carré) s'il est le produit de deux nombres successifs ***k*** *et* ***(k+1)***, c'est-à-dire ***N = k*(k+1)***. Il est aussi la somme des ***k*** premiers nombres pairs.

Par exemple, les entiers 0 (=0*1) , 2 (=1*2), 6 (=2*3), 12 (=3*4), 20, 30 et 42 sont des nombres proniques.

Pour chercher la valeur de ***k***, on calcule la somme des ***k*** premiers nombres pairs jusqu'à ce que cette somme devienne supérieure ou égale à ***N***.

Exemples : - Si **N = 20** alors S = 2+4+6+8 **= N ➔ N est pronique et k = 4 (20 = 4*5).**

- Si **N = 23** alors S = 2+4+6+8+10 = 30 > **N ➔N n'est pas pronique et k = 5 (23 < 5*6).**

Q1) En respectant cette définition, écrire une fonction **Pronique(N)** qui permet de vérifier si un entier **N** est pronique ou pas.

Q2) Ecrire l'algorithme principal qui permet d'afficher les **M** petits nombres proniques.

Exemple : Si M = 10 alors les 10 petits nombres proniques sont : 0, 2, 6, 12, 20, 30, 42, 56, 72, 90.

Exercice 4 : Fonctions numériques

Q1) Factorielle : Ecrire une fonction **Facto(n)** qui permet de calculer **n ! = 1 * 2 * 3 * …*n.**

Q2) Combinaisons : Ecrire une fonction **Combin (n, p)** qui permet de calculer le nombre de combinaisons défini par $C_n^p = \dfrac{n!}{p!*(n-p)!}$. **n** et **p** sont deux entiers positifs tels que **p ≤ n**.

Q3) Puissance : Ecrire une fonction **Puiss(x, n)** qui renvoie **x^n= x * x * … * x**(**x** réel et **n** entier positif).

Q4) Exponentielle : Ecrire une fonction **Expn(x, n)** qui permet de calculer la valeur approchée $e^x \cong 1+\dfrac{x}{1}+\dfrac{x^2}{2!}+\dfrac{x^3}{3!}+...+\dfrac{x^n}{n!}$ (**x** est un nombre réel et **n** un nombre entier positif).

Exercice 5 : Récursivité

Q1) Factorielle : Ecrire une fonction **FactoRec(n)** qui permet de calculer **n ! = n * (n-1) !**

(On sait que 0 ! = 1 ! = 1)

Q2) Puissance : Ecrire deux fonctions récursives pour calculer **x^n**.

- La première **PuissRec(x, n)** en utilisant la définition suivante :

$$x^n = \begin{cases} 1 & Si \;\; n=0 \\ x^{n-1} \times x & Si \;\; n>0 \end{cases}$$

- La deuxième **PuissRec2(x, n)** en utilisant la définition suivante :

$$x^n = \begin{cases} 1 & Si \;\; n=0 \\ x^{n/2} \times x^{n/2} & Si \;\; n>0 \; et \;\; n \; est \; pair \\ x^{(n-1)/2} \times x^{(n-1)/2} \times x & Si \;\; n>0 \;\; et \;\; n \;\; est \; impair \end{cases}$$

Q3) (Optionnelle) L'algorithme d'Euclide permettant de calculer le **PGCD** (Plus Grand Commun Diviseur) de deux entiers strictement positifs **A** et **B** tels que **A** > **B** est défini comme suit :

$$PGCD(A,B) = \begin{cases} PGCD(B,\; A \bmod B) & Si \; B \neq 0 \\ A & Si \; B=0 \end{cases}$$

Ecrire une fonction récursive permettant de déterminer le **PGCD** de**A** et **B**.

Q4) (Optionnelle) Suite de Fibonacci : Ecrire une fonction récursive **Fibo(n)** permettant de calculer le **n^ieme^** terme de la suite de **Fibonacci** définie par : $U_n = \begin{cases} 0 & Si \; n=0 \\ 1 & Si \; n=1 \\ U_{n-1}+U_{n-2} & Si \; n \geq 2 \end{cases}$

Q5) Pour un entier positif N, écrire une fonction <u>itérative</u> puis une autre <u>récursive</u> qui permet :

- Calculer le nombre de chiffres de N.
- Calculer la somme des chiffres de N.
- Calculer le produit (la multiplication) des chiffres de N.

<u>Remarque</u> : Si le nombre N contient un zéro, dès que le chiffre 0 (zéro) est rencontré, le produit devient nul (égal à zéro), il faut s'arrêter.