

Ministère de L'Education Et De La Formation C.R: Kairouan LYCEE OKBA ===== DEVOIR DE SYNTHESE N : 1	Classe : 4^{eme} S.I	
	Epreuve : Programmation	
	Enseignant : ALOUINI.H	
	Coefficient : 3	
	Durée: 3 Heures	Année 2018/2019

Nom & prénom : Numéro :

Le sujet comporte 4 pages.

EXERCICE N 1(3 pts) :

Mettre dans chaque case, la lettre V si la proposition est correcte ou la lettre F dans le cas contraire.

1. La fonction verif permet de verifier si un entier N est premier ou non

<input type="checkbox"/> 0-Def fn verif(N :entier) :boolean 1-S←0 2-Pour I de 2 a N faire si n mod i =0 then s←s+1 Fin si fin pour Verif←s=1	<input type="checkbox"/> Def fn verif(N :entier) :boolean S←0 2-Pour I de 1 a N faire si n mod i =0 then s←s+1 Fin si fin pour 3- Verif←s=2 4-Fin verif	<input type="checkbox"/> Def fn verif(N :entier) :boolean 1- S←0 2- Pour I de 2 a N-1 faire si n mod i =0 then s←s+1 Fin si fin pour 3-Verif←s=0
---	---	---

2- Chercher l'élément le plus grand dans un tableau

- | | |
|--------------------------|--|
| <input type="checkbox"/> | Est un problème récurrent d'ordre 1 |
| <input type="checkbox"/> | Peut être traité d'une façon itérative |
| <input type="checkbox"/> | Peut être traité d'une façon récursive |

3- la fonction conv permet de convertir un entier de la base 10 vers la base 2

<input type="checkbox"/> 0- Def fn conv(N : entier) :chaine 1- R←'' 2- Repeter R←R+chr(N mod 2+ord('0')) N←N div 2 jusqu'a N=0 3- Conv←R 4- Fin conv	<input type="checkbox"/> 0-Def fn conv(N : entier) :chaine 1-Si N=0 alors conv←'' Sinon conv←chr(N mod 2+ord('0'))+conv(N div 2) fin si 2-Fin conv	<input type="checkbox"/> 0-Def fn conv(N : entier) :chaine 1- R←'' 2-Repeter R←chr(N mod 2+ord('0'))+R N←N div 2 jusqu'a N=0 3-Conv←R 4- Fin conv
---	--	--

4- la procédure triangle permet de remplir une matrice M par le triangle de pascal

<input type="checkbox"/> 0- Def proc triangle(var M :mat,N: entier) 1- Pour I de 1 a N faire M[i,i]←1 M[1,i]←1 pour J de 2 a I-1 faire M[i,j]←M[i-1,j-1]+M[i-1,j] fin pour fin pour 2- Fin triangle	<input type="checkbox"/> 0-Def proc triangle(var M :mat,N: entier) 1- Pour I de 1 a N faire M[i,i]←1 pour J de 1 a I-1 faire M[1,i]←1 si j<>1 alors M[i,j]←M[i-1,j-1]+M[i-1,j] fin si fin pour fin pour 2- Fin triangle	<input type="checkbox"/> Def proc triangle(var M :mat,N: entier) Pour I de 1 a N faire M[1,i]←1 pour J de 2 a I faire M[i,i]←1 si i<>j alors M[i,j]←M[i-1,j-1]+M[i-1,j] fin si fin pour fin pour 2- Fin triangle
---	---	--

EXERCICE N 2 : (3.5pts)

Par définition, un nombre sympathique est un nombre entier naturel : qui est une puissance de 2 (2^n avec $n \in \mathbb{N}^*$) exemple : 8,16, 64, . . .

Afin de déterminer si un entier n est sympathique ou non un élève propose ce programme pascal :

Function sympathique(n:integer):boolean;

Var

.....

begin

a:=1;

Repeat

*a:=a*2;*

until a>=n;

if a=n then Sympatique:=true

else

Sympatique:=false

end;

Questions

1- Compléter la partie déclarative de cette fonction.

2- Exécuter à la main la fonction, en donnant le résultat d'exécution pour les deux cas suivant :

n =32

n =15

3- Par définition, un nombre **Super-sympathique** est un nombre entier naturel : qui ne se compose que de chiffres représentant une puissance de 2 (2^n avec $n \in \mathbb{N}^*$)

Exemple : nombres Super-sympathiques inférieurs à 100 : 2; 4; 8; 22; 24; 28; 42; 44; 48; 82; 84; 88

Ecrire algorithme d'un sous programme qui permet de vérifier si un nombre est super-sympathiques de 5 chiffres.

.....
.....
.....
.....
.....
.....
.....
.....
.....
.....

Exercice N : 3 (4pts)

Soit la fonction récursive suivante écrite en Pascal, qui permet de vérifier la primalité d'un entier naturel non nul (Premier ou non).

```

FUNCTION Premier (N, X : LONGINT) : ..... ;
BEGIN
IF X > N DIV 2 THEN
    Premier: = True
ELSE
    IF N MOD X = 0 THEN
        Premier: = False
ELSE
        Premier: = Premier ( N , X + 1 ) ;
END;

```

- 1°) Compléter le type de cette fonction.
- 2°) Ecrire cette fonction avec une version itérative.

.....

.....

.....

.....

.....

.....

.....

.....

.....

3°) En mathématiques, **la constante de Brun (B) des nombres premiers jumeaux** (ou plus simplement constante de Brun) est la somme de la série **des inverses des nombres premiers jumeaux**, c'est-à-dire des couples de nombres premiers distants de 2.

On demande d'écrire l'analyse d'une fonction intitulée « **Brun** » permettant de retourner une valeur approchée de la constante de **Brun (B)** en utilisant :

La formule suivante :

$$B = \left(\frac{1}{3} + \frac{1}{5}\right) + \left(\frac{1}{5} + \frac{1}{7}\right) + \left(\frac{1}{11} + \frac{1}{13}\right) + \left(\frac{1}{17} + \frac{1}{19}\right) + \left(\frac{1}{29} + \frac{1}{31}\right) + \dots$$

La fonction « **Premier** » déjà définie à la question n° 1.

N.B. : Le calcul s'arrête quand la différence entre deux sommes successives devient inférieure ou égale à une erreur « **Epsilon** » (Epsilon est donné comme paramètre).

.....

.....

.....

.....

.....

.....

.....

.....

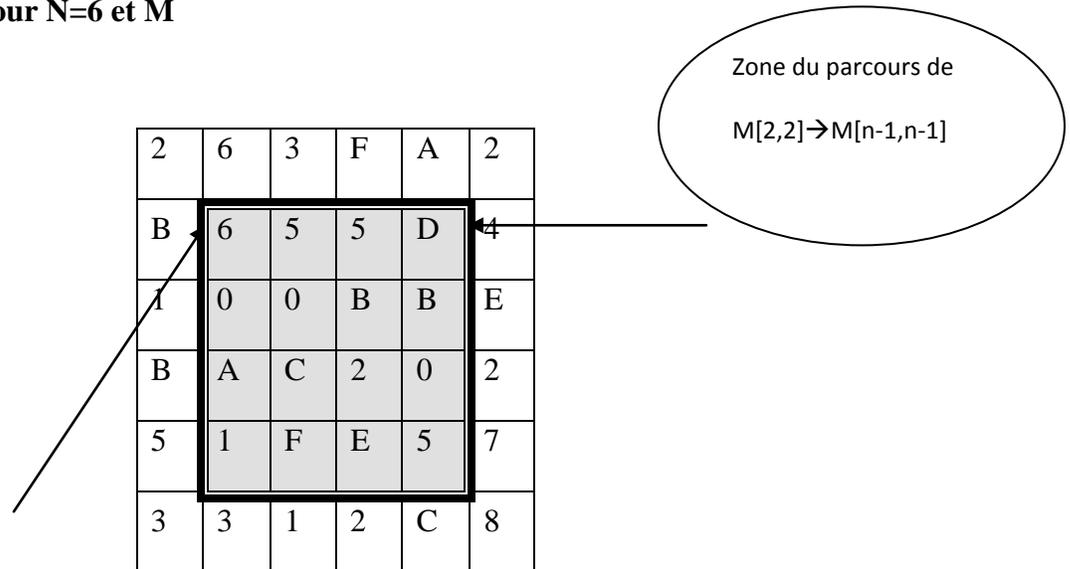
.....

Problème : (9.5pts)

On vous demande d'écrire une analyse modulaire d'un programme qui permet de :

- Remplir aléatoirement une matrice carrée M de taille N ($4 < N < 30$) par des caractères de la base 16 (0..9,A..F).
- Remplir a partir de la matrice M un fichier type d'enregistrement nommée **code.doc** composée de deux champs comme suit :
 - **NL** : entier qui contient la somme des indices d'un élément de M exemple l'élément M[2,2] donc NL contient 4 car $2+2=4$
 - **code** : contient pour chaque élément de M (on commence par M[2,2] jusqu'à M[N-1,N-1]) le groupement voisin de lui trié dans l'ordre croissant selon les colonnes on utilisant le trie par insertion.

Exemple pour N=6 et M



On prend les voisins de chaque M[i,j]. exemple pour M[2,2] on aura NL=2+2=4

2	6	3
B	6	5
1	0	0

Trié croissant selon les

1	0	0
2	6	3
B	6	5

colonnes

ON OBTIENT LE FICHIER **CODE.DOC** COMME SUIV

NL	4	5	6	7	5			
CODE	100263B65	0660355BF	0535BFABD	5BFABD24E	...			