

16/20
UE MG102

NOM & PRENOM :

Aucun document autorisé & calculatrice non autorisée.

Répondre sur le sujet.

Durée : 1 heure 30 minutes

Question 1 On dit que 2 nombres entiers naturels non nuls a et b sont « copremiers » si leur seul diviseur commun est 1. Exemple : 9 et 10 sont copremiers car le seul entier qui divise à la fois 9 et 10 est 1. Par contre 15 et 20 ne sont pas copremiers car ils sont tous les deux divisibles par 5.

Écrire deux algorithmes vérifiant si deux entiers naturels non nuls a et b sont copremiers.

Le résultat est le booléen true si a et b sont copremiers, sinon le résultat est le booléen false.

1. le premier algorithme copremiers doit utiliser l'itération POUR

2. le second algorithme copremiers doit utiliser l'itération TANTQUE et s'arrêter dès qu'un diviseur commun à a et b autre que 1 est trouvé.

Nom: copremiers 1

Données : a et b nombres tel que $a \geq b$ Résultat: true si a et b sont copremiers et false sinon.

Début:

Pour i allant de 2 à a fairesi $a \bmod i = 0$ et $b \bmod i = 0$ alors false

sinon true

fin si

 $i = i + 1$

Fin POUR

Fin

2/3

Nom: copremiers 2

Données: a et b nombres tel que $a \geq b$ Résultat: true si a et b sont copremiers et false sinon.Début: $i = 2$ si $a \bmod i = 0$ et $b \bmod i = 0$ alors falsesinon Tant que $i \leq a$ $P = a \bmod i$ et $T = b \bmod i$ si $P = T = 0$ alors false

sinon True

fin si et Fin tant que

fin

idem.

2/3

Question 2 Ecrire un algorithme nbNegatifs qui étant donné un tableau de nombres T, calcule le nombre d'éléments négatifs de T

Exemple : si Ta est la tableau 4 | -3 | -1 | 9 | 5 | -8 alors nbNegatifs(Ta)=3

Nom: nbNegatif
 Données: T: tableau de nombre
 Resultat: nombre, le nombre total d'élément négatif
 Debut:
 Pour i allant de 1 à taille(T)-1 faire
 Si T[i] < 0 alors 0 + nbNegatif(T[i+1])
 sinon
 Si T[i] < 0 alors 1 + nbNegatif(T[i+1])
 sinon 0 + nbNegatif(T[i+1])
 Fin si
 Fin si
 Fin

3/5/4

Question 3 : Que fait l'algorithme ci-dessous ? Que désigne selon vous n et m ? (ref2-2)

A noter : A,B,C sont des tableaux, Afini et Bfini des booléens, et ia,ib,ic des indices pour parcourir les tableaux. Redim est une fonction permettant de modifier la taille d'un tableau (agrandir ou diminuer).

DEBUT

(...)

Afini ← faux

Bfini ← faux

ia ← 0

ib ← 0

ic ← 0

TantQue Non(Afini) ou Non(Bfini)

ic ← ic + 1

Redim C[ic]

Si Afini ou A[ia] > B[ib] Alors

C[ic] ← B[ib]

ib ← ib + 1

Bfini ← ib > n

Sinon

C[ic] ← A[ia]

ia ← ia + 1

Afini ← ia > m

FinSi

FinTantQue

FIN

Question 4 : identifier des algorithmes (ref3-1)

Des trois algorithmes ci-dessous, lequel est le tri à bulle, le tri par sélection et le tri par insertion ?

```

Pour i de 0 à n-2 Faire
    min ← i
    Pour j de i+1 à n-1 Faire
        Si tab[j] < tab[min] Alors min ← j FinSi
    FinPour
    Echanger tab[i] et tab[min]
FinPour
  
```

ALGORITHME DE TRI : *Par sélection*

```

Pour i de 1 à n-1 Faire
    val ← tab[i]
    j ← i-1
    TantQue j >= 0 et tab[j] > val Faire
        tab[j+1] ← tab[j]
        j ← j - 1
    FinTantQue
    tab[j+1] ← val
  
```

ALGORITHME DE TRI : *Par insertion*

```

Pour i de 0 à n-2 Faire
    Pour j de 0 à n-2-i Faire
        Si tab[j+1] < tab[j] et
        Alors échanger tab[j+1] et tab[j]
        FinSi
    FinPour
FinPour
  
```

ALGORITHME DE TRI : *à bulle*

Question 5 : corriger un algorithme qui présente des fautes (ref2-3)

Pour l'algorithme suivant, donnez son but ? Et corrigez les erreurs directement sur l'algo (rayer, modifier,... et donnez la bonne écriture).

Algorithme : mystere

Données : ta : Tableau de Nombre, k : caractère entier

Résultat : Tableau de Nombre, mystere...

Variation : tres : Tableau de nombre de taille(ta)-1 éléments

debut


```

pour i de 0 à k faire
    tres[i]:=ta[i]
finTantQue fin Pour
pour i de k+1 à taille(ta)-1 faire
    tres[i-1]=ta.(i) tres[i-1]=ta[i]
finPour

```

2/2

Le résultat est : ta

fin

BUT DE L'ALGORITHME former un tableau dont le début les $k^{\text{ème}}$ éléments appartiennent à ta et les derniers éléments appartiennent à tres

Question 6 : détection de collision d'objets dans les jeux vidéos (ref5-7)

Construire un algorithme permettant de résoudre le problème suivant :

□ Données : deux cercles C1 et C2 , chacun étant spécifié par l'abscisse et l'ordonnée de son centre, et son rayon (soit 3 variables par cercle).

□ Résultat : "vrai" si l'intersection de C1 et C2 n'est pas vide (les cercles sont en collision), et "faux" sinon.

En mathématiques: l'intersection de C_1 et C_2 n'est pas vide si et seulement si la distance entre (x_2, y_2) et (x_1, y_1) qui sont les centres respectifs de C_1 et C_2 est égale ou plus petit que $r_1 + r_2$;

$$\sqrt{(x_1 + x_2)^2 + (y_1 + y_2)^2} \leq r_1 + r_2$$

JK

2