Lycee Charlemagne | MPSI2 | Annee 2021/22

Info complexité

Jeudi 18 novembre

La MPSI2 est faite de 50 élèves. Chaque élève est une liste contenant dans l'ordre

0	nom
1	note de maths
2	note de physique
3	abscisse
4	ordonnée

(les coordonnées de références sont la salle 210)

Cette liste de listes est triée par notes de maths croissantes, de `['Sucri', 1.2, 2.6, 0, 0]`
à `['Choquet', 20.0, 1.2, 1, 2]`.

On généralisera à N élèves.

Si il faut 1 seconde par opération,							
Complexité	$\ln(N)$	N	$N.\ln(N)$	N^2	N^3	2^N	$N!$
avec $N = 10$	2 secondes	10 secondes	23 secondes	1 minute et demi	un quart d'heure	un quart d'heure	un mois et demi
avec $N = 20$	3 secondes	20 secondes	1 minute	7 minutes	2 heures	12 jours	1 miliard de siècles
avec $N = 40$	3 secondes et demi	40 minutes	2 minutes et demi	trois quart d'heure	17 heures	35 siècles	10^{40} siècles
avec $N = 60$	4 secondes	1 minute	4 minutes	1 heure	deux jours et demi	des milliards d'années	10^{72} siècles
pour chacune des complexité $C(n)$, exprmiez $C(2.n)$ à l'aide de $C(n)$							
$C(2.n)$							

.

Associez à chaque objectif, son algorithme et sa complexité.			
	objectif	algorithme	complexité
A	* calculer la moyenne des notes de physique		
B	* trouver toutes les configurations pour asseoir en classe les 50 élèves		
C	* trouver l'élève qui habite le plus loin de Charlemagne		
D	* trouver les deux élèves qui habitent le plus près l'un de l'autre		
E	* savoir si au moins un élève a eu 9 en maths		
F	* savoir si au moins un élève a eu au moins 9 en maths		
G	* connaître la médiane des notes de maths		
H	* trouver la superficie du plus petit polygone contenant tous les élèves de la classe		
I	* connaître le rang en physique du meilleur élève en maths		
J	* donner la liste triée des notes de physique		
K	* donner la liste sans doublon des notes de physique		
L	* donner la liste de tous les trinômes qu'on peut créer		
Complexités $O(1)$, $O(N)$, $O(N.\ln(N))$, $O(n^2)$, $O(N^3)$, $O(N!)$			

```
def Scooby(Classe) : #list -> float
....S = 0
....for Eleve in Classe :
........S += Eleve[2]
....return(S/len(Classe))
```

```
def Vera(Classe) : #list -> int
....Note = Classe[-1][1]
....c = 0
....for Eleve in Classe :
........if Eleve[2] < Note :
............c += 1
....return(c)
```

```
def Daphne(Classe) : #list -> float
....n = len(Classe)
....if n%2 == 1 :
........return(Classe[n//2][1])
....else :
........return((Classe[n//2][1]+Classe[1+n//2][1])/2)
```

```
def Velma(Classe) : #list -> boolean
....return(Classe[-1][1] >= 9)
```

```
def Samy(Classe) : #list -> string
....Maxi = 0
....EleveLoin = " "
....for Eleve in Classe :
........d = Eleve[3]**2+Eleve[4]**2
........if d > Maxi :
............Maxi = d
............EleveLoin = Eleve[0]
....return(Eleve[0])
```

```
def Scoubidou(L) : #list -> boolean
....#print(L)
....if len(L) == 1 :
........return(L[0] == 9)
....m = len(L)//2
....if L[m] == 9 :
........return(True)
....if L[m] < 9 :
........return(Scoubidou(L[m+1 : ]))
....if L[m] > 9 :
........return(Scoubidou(L[ : m]))
```

```
def Fred(L) : #list -> string, string
....Mini = float("inf")
....Qui = [0, 0']
....for j in range(len(L)) :
........for i in range(j) :
............d = (L[i][3]]-L[j][3])**2+(L[i][4]]-L[j][4])**2
............if d < Mini :
................Mini = d
................Qui = [i, j]
....return(L[i][0], L[j][0])
```

```
def Shaggy(n) : #int -> list of list
....if n == 0 :
........return([[ ]])
....L = Shaggy(n-1)
....LL = [ ]
....for k in range(n) :
........Lk = [lis[ : ] for lis in L]
........for lis in Lk :
............lis.insert(k, n)
............LL.append(lis)
....return(LL)
```

.

Question subsidiaire : expliquez moi ce qu'une Totally Spies fait dans un dessin animé de chez Hanna-Barbera

.

```
def Scoubidur(Classe) : #list -> list
....L = [ ]
....for Eleve in Classe :
........if not(Eleve[2] in L) :
............L.append(Eleve[2])
....return(L)
```

```
def Clover(Classe) : #list -> list
....L = [Eleve[2] for Eleve in Classe]
....n = len(L)
....for i in range(n) :
........for j in range(i+1 , n) :
............if L[i] < L[j] :
................L[i], L[j] = L[j], L[i]
....return(L)
```