

Interrogation 1 : récurrence et itérations

Durée : 25 minutes.

Nom :

Prénom :

Attention : le sujet est recto-verso, n'oubliez pas de tourner la page.

1 Programmation générique

Complétez le code suivant pour que la fonction `somme_int` réalise la somme de deux entiers passés en paramètre via leurs adresses génériques.

```
int somme_int(const void * a, const void * b) {  
  
    /* ← remplir ici */  
  
}  
  
int main() {  
    int a = 5 ;  
    int b = 8 ;  
    printf("%d\n", somme_int(&a,&b)) ; /* -> 13 */  
    printf("%d\n", somme_int(&a,&a)) ; /* -> 10 */  
    printf("%d\n", somme_int(&b,&b)) ; /* -> 16 */  
    return 0 ;  
}
```

2 Boucles pour

Calculez dans les algorithmes suivants le nombre d'appels à `instruction()` en fonction de n . Vous considérerez que dans une boucle « **pour** i allant de a à b », les bornes a et b de la boucles sont incluses : la première itération aura $i = a$ et la dernière $i = b$. Plus que l'expression exacte et simplifiée du nombre d'appels en fonction de n , c'est le raisonnement qui permet d'aboutir à ce résultat qui est important. Si vous ne parvenez pas à simplifier une formule, indiquez la donc quand même.

données : $n \geq 5$: un entier
Algorithme
a) $\left[\begin{array}{l} \text{pour } i \text{ allant de } 5 \text{ à } n \text{ faire} \\ \quad \left[\text{instruction}() \right] \end{array} \right.$

données : $n \geq 1$: un entier
Algorithme
b) $\left[\begin{array}{l} \text{pour } i \text{ allant de } 1 \text{ à } n \text{ faire} \\ \quad \left[\begin{array}{l} \text{pour } j \text{ allant de } i \text{ à } n \text{ faire} \\ \quad \left[\text{instruction}() \right] \end{array} \right] \end{array} \right.$

3 Récurrence

Prouvez par récurrence que pour tout $n \geq 0$, $\sum_{k=0}^n 2^k = 2^{n+1} - 1$