1	
2	
3	

Université de La Réunion – FST

Licence d'informatique – L3 – Mai 2024

Contrôle terminal de l'U.E. *vérification et complexité*

Durée : 1h (1h20 si tiers temps) – sans document ni moyen électronique

Répondre uniquement dans les cadres prévus à cet effet. La gestion de l'espace fait partie de l'épreuve.

Nom :	Signature :
Prénom(s) :	

Exercice 1 (5 •) Remplissez le tableau ci-dessous à raison de deux items par ligne en assurant la cohérence de chaque ligne. On s'intéresse à la complexité dans le pire des cas.

Complexité	$\Theta(\quad)$ ou $O(\quad)$	*Algorithme*
exponentielle		
	$\Theta(n^2)$	
linéaire		
		Recherche dichotomique d'un élément dans un tableau trié de taille n
quasi-linéaire		

Exercice 2 (10 •) On considère le programme suivant dont on souhaite montrer la correction totale :

Entrées : a et b, deux entiers naturels
Sortie : le produit de a par b

```
x:entier := 0;
y:entier := a;
(1) tant que y <> 0 faire
    (2) x := x+b;
        y := y-1;
    (3)
(4) retourner x
```

(2 •) Montrez que $y \geq 0$ est un invariant inductif au point de programme (1).

(2 •) Montrez que $x + y * b = a * b$ est un invariant inductif au point de programme (1).

(1 •) Montrez que si la boucle `tant que` termine, le programme retourne la valeur $a * b$ au point de programme (4).

Nom : Signature :

Prénom(s) :

(2 •) Montrez que le programme termine en validant une fonction de rang pour la boucle.

(2 •) Déterminez et justifiez la complexité du programme.

(1 •) Proposez une relaxation non-triviale de la pré-condition (*i.e.*, une pré-condition plus générale que *a et b, deux entiers naturels*) telle que les raisonnements justifiant les questions précédentes restent valides. Argumentez.

Exercice 3 (5 •) Dans cet exercice, on mesure la complexité d'un programme par le nombre d'affectations portant sur une variable de type *élément de tableau d'entiers*. D'autre part, `a` est une variable de type tableau d'entiers de dimension adéquate et `n` le paramètre entier utilisé pour définir la complexité. Donnez avec une brève justification (quelques mots) la complexité en $\Theta()$ des fragments de code suivants :

```
a[1] := 1
```

```
pour i de 1 à n faire
        a[i] := 1
```

```
pour i de 1 à n faire
        pour j de 1 à n faire
                a[i,j] := 1
```

```
pour i de 1 à n faire
        pour j de 1 à i faire
                a[i,j] := 1
```

```
pour i de 1 à n faire
        pour j de 1 à n faire
                pour k de 1 à n faire
                        a[i,j,k] := 1
```