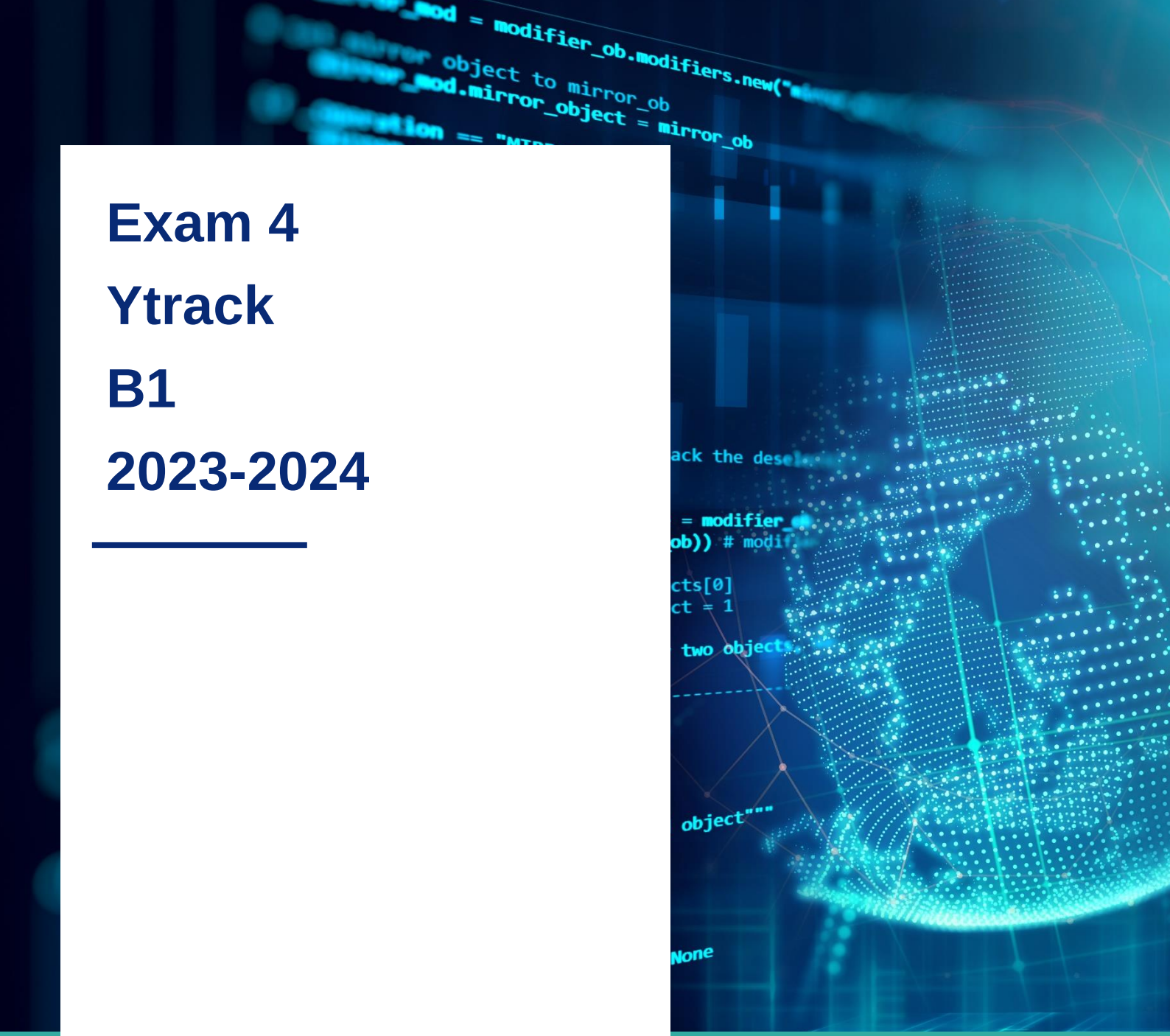
Exam 4

Ytrack

B1

2023-2024



7 AOUT

Contexte :

Vous avez 4h
Vous avez le droit à internet.
Vous avez le droit d’aller voir vos anciens codes.

1 exercice réussi : 5 points
2 exercices réussi : 10 points
3 exercices réussi : 14 points
4 exercices réussi : 17 points
5 exercices réussi : 20 points

Un exercice est considéré comme réussi s’il renvoie la valeur adéquate au test pendant la correction, quel que soit sa méthode de résolution.

Les exercices peuvent être réalisé dans n'importe quel ordre.

Si un exercice rendu, réussi ou non, comporte des ressemblances que nous considérons comme flagrante avec des ressources en ligne, ou avec des exercices rendu par d'autres étudiants, alors il sera considéré comme de la triche.
Toute utilisation de ChatGPT observée par un surveillant sera considérée comme de la triche

Toute triche résultera en a 0 pour l'étudiant. Non négociable.

1.Ft_coin

Créer la fonction Ft_coin.

Ft_coin a 2 paramètres :
-coins correspond aux différentes valeurs des pièces que vous possédez.
-amount correspond à la valeur souhaitée avec les pièces possédées.

La fonction retourne le plus petit nombre de pièces nécessaire pour atteindre la valeur amount selon les pièces précisées dans coins.

Si le nombre n'est pas atteignable, le retour doit être -1.

Prototype :

```
func Ft_coin(coins []int, amount int) int {
    ...
}
```

Test:

```
func main() {
    Ft_coin([]int{1, 2, 5}, 11) // resultat :  3 car (11 == 5 + 5 + 1)
    Ft_coin([]int{2}, 3)        // resultat : -1
    Ft_coin([]int{1}, 0)        // resultat :  0
}
```

2.Ft_missing

Créer la fonction Ft_missing.

Ft_ missing a 1 paramètre :
-num contient n nombres différents compris entre [0,n].

La fonction retourne l'unique nombre manquant dans l'intervalle [0,n]

Prototype :

```
func Ft_missing(nums []int) int {
    ...
}
```

Test:

```
func main() {
    Ft_missing([]int{3, 1, 2})                  // resultat : 0
    Ft_missing([]int{0, 1})                     // resultat : 2
    Ft_missing([]int{9, 6, 4, 2, 3, 5, 7, 0, 1}) // resultat : 8
}
```

3.Ft_non_overlap

Créer la fonction Ft_non_overlap.

Ft_non_overlap a 1 paramètre :
-intervals est un tableau d'intervalle.

La fonction retourne le plus petit nombre d'intervalle à retirer pour que les intervalles restant ne se superpose pas.

Prototype :

```
func Ft_non_overlap(intervals [][]int) int {
    ...
}
```

Test:

```
func main() {
    Ft_non_overlap([][]int{{1, 2}, {2, 3}, {3, 4}, {1, 3}}) // resultat : 1
    // pour que les intervalles soient tous des intervalles qui ne se superpose pas,
    // il suffit de d'enlever qu'un seul intervalle, [1,3]
    Ft_non_overlap([][]int{{1, 2}, {2, 3}})                 // resultat : 0
    Ft_non_overlap([][]int{{1, 2}, {1, 2}, {1, 2}})         // resultat : 2
}
```

4.Ft_profit

Créer la fonction Ft_profit.

Ft_profit a 1 paramètre :
- prices et un tableau de prix ou prices[i] est le prix d'un bien au i’eme jours.

La fonction retourne le plus grand bénéfice possible pour le bien, au sein de ces jours, en effectuant de l'achat revente.

Les prix seront toujours positifs.

Prototype :

```
func Ft_profit(prices []int) int {
    ...
}
```

Test:

```
func main() {
    Ft_profit([]int{7,1,5,3,6,4}) // resultat : 5
    // si on achète au jour 1, nous payons 1,
    // et si nous le vendons au 4eme jour, nous gagnons 6, le bénéfice est 6-1
    Ft_profit([]int{7,6,4,3,1}) // resultat : 0
}
```

5.Ft_max_substring

Créer la fonction Ft_max_substring.

Ft_max_substring a 1 paramètre :
- s est une chaine de caractère

La valeur de retour est la longueur de la plus grande sous chaine de s possédant des caractères non répétés au sein de s.

Prototype:

```
func Ft_max_substring(s string) int {
    ...
}
```

Test:

```
func main() {
    Ft_max_substring("abcabcbb")    // resultat : 3
    // "abc" est la plus grande sous chaine composé de caractère diffèrent
    Ft_max_substring("bbbbb")       // resultat : 1
    // "b" est la plus grande sous chaine
}
```

6. Ft_min_window

Créer la fonction Ft_min_window

Ft_min_window a 2 paramètres :
- s est une chaine de caractère
- t est une chaine de caractère

La valeur de retour est la plus petite séquence compris dans "s" de tel sorte a ce que chaque caractère de "t" y soit compris.

Si deux séquences on la même taille, retourné la séquence arrivant en première dans la chaine de caractère s.

Prototype:

```
func Ft_min_window(s string, t string) string {
    ...
}
```

Test:

```
func main() {
    Ft_min_window("ADOBECODEBANC", "ABC") // resultat : "BANC"
    Ft_min_window("a", "aa")              // resultat : ""
}
```