*INFORMATIQUE Classe de Seconde* *Nom :……..*

# *Contrôle n° 1*

## *Première partie (répondre sur cette feuille)*

***Sur les langages de programmation et la programmation***

***1.*** Expliquer la différence entre un langage dit de bas niveau et un langage évolué.

……

……

……

……

***2.*** Expliquer la différence entre un langage interprété et un langage compilé.

……

……

……

……

***3.*** Les deux premiers langages évolués sont FORTRAN et LISP (dont DrRacket est un dialecte) et les noms de ces deux langages sont des contractions de deux expressions anglaises : quelles sont ces deux expressions ?

……

……

***4.*** Quand on cherche à écrire une fonction, avant d'écrire quoi que ce soit, quelle est la première chose à faire ?

……

……

***5.*** Expliquer en quelques mots :

a) ce qu'est une fonction récursive

……

……

……

……

***b)*** ce qu'est une fonction récursive terminale

……

……

……

***c)*** ce qu'est une fonction récursive non terminale

……

……

***6. Soit la fonction suivante :***

```
(define (inconnu  n)
     (if (<= n 2 )
          n
          (if (even? n)        ; anglais    even = pair
              (inconnu (div2 n))    ; div2  = la division entière par 2
              (inconnu (1+ (* 3 n))))))
```

Faire tourner « à la main » (*sur cette feuille*) cette fonction sur l'exemple proposé : (Faire d'abord au brouillon) (*en clair, faire une trace de la fonction comme il a été vu en cours – respecter l'indentation des appels !- et bien faire apparaître les résultats intermédiaires en correspondance avec les appels correspondants):*

```
(inconnu 6)
```

⟶ ?

⟶ ?

Résultat final : …….

Que calcule cette fonction ?

……..

Cette fonction est –elle récursive terminale ?

……..

***7. Soit la fonction suivante :***

```
(define (inconnu2 a b)
     (if (zero? b)
         a
          (inconnu2 b
          (remainder a b))))
```

***Information :*** la fonction ***remainder*** (qui est une fonction prédéfinie de Scheme) retourne le reste de ***la division entière*** de a par b

***Exemples***

*(remainder 15 5)retourne 0*

*(remainder 23 4)retourne 3*

Faire tourner « à la main » (*au brouillon cette fois*) cette fonction en prenant :

a) a = 45 b = 10 Résultat : …….

b) a = 206 b = 4 Résultat : …….

c) a = 77 b = 6 Résultat : …….

et dans chaque cas, préciser le résultat retourné par la fonction.

Que calcule cette fonction ?

……

# *Seconde*
# *Interrogation Informatique*

## *Deuxième partie (sur machine)*.

***Au début de la fenêtre de définitions, taper*** :

`;  votre nom  et votre classe` (*ne pas oublier* ***le point virgule*** *au début*)

Taper rapidement les fonctions suivantes en début de fenêtre de définition

```
(define (1+ x)          (define (1- x)          (define (mul2 n)        (define (div2 n)
 (+ x 1))                (- x 1))                (* 2 n))                 (quotient n 2))
```

et ***sauvegarder immédiatement la fenêtre de définition*** dans un dossier provisoire ou sur le bureau
fichier à envoyer à mlagouge@wanadoo.fr ***à la fin du DST*** puis effacer le fichier sur l'ordinateur

***Exercice 1 : fonction somme_entiers***

Ecrire la fonction ***somme_entiers*** qui :

- a comme argument un entier positif : n
- calcule la somme S = 0 + 1 + 2 + 3 + ... + n

*Exemple :* `(somme_entiers 5)` *(0 + 1 + 2 + 3 + 4 + 5)*

```
15
```

***Exercice 2 : fonction somme_impairs***

Ecrire la fonction ***somme_impairs*** qui

- a comme argument un entier positif ***impair***: n
- calcule la somme des entiers impairs jusque n  S = 1 + 3 + 5 + 7 …….+ n

*Exemple :* `(somme_impairs 5)`

```
9
```

Tester `(somme_impairs 19)` `(somme_impairs 39)` `(somme_impairs 99)`

***Donner les résultats dans le fichier (en faisant précéder le résultat par le point virgule) comme ci-dessous***

```
; (somme_impairs 19)
; ????????
```

***Exercice 3 : fonction somme_pairs***

Ecrire la fonction ***somme_pairs*** qui

- a comme argument un entier positif ***pair***: n
- calcule la somme des entiers impairs jusque n  S = 0 + 2 + 4 + 6 …….+ n (avec n = 2x k puisque pair !)

*Exemple :* `(somme_pairs 10)` *(0 + 2 + 4 + 6 + 8 + 10)*

```
30
```

***Exercice 4 : fonction Fibonacci***

La suite de Fibonacci est définie par la relation suivante :

Fib(0) = 0  Fib(1) = 1  et  Fib (n) = Fib(n-1) + Fib (n-2)

1) Calculer rapidement (sur brouillon) Fib(2) = ……  Fib(3) = …… Fib(10) = ……

*Mettre les résultats dans la fenêtre de définition en précédant les réponses de ;* -point virgule - soit par ex

```
; fib(2) = ????        Ne pas oublier le point virgule en début de ligne
```

2) écrire la fonction ***Fib*** en Scheme

3) A l'exécution de la fonction, que valent (***Fib*** 25) et (***Fib*** 30) ? (*Mettre les résultats dans la fenêtre de définition*)

4) que se passe-t-il si on veut faire calculer (***Fib*** 1000) ? (*Mettre votre réponse dans la fenêtre de définition)*

***Bonus Exercice 5 : fonction somme_puis2***

Ecrire la fonction ***somme_puis2*** qui

- a comme argument un entier ***positif***: n
- calcule la somme des puissances de 2 jusque $2^n$  S = $2^0 + 2 + 2^2 + 2^3$ …….+ $2^n$  *(rappel : $2^0 = 1$)*

*Exemples :* (***somme_puis2*** 4) (***somme_puis2*** 7) (***somme_puis2*** 9)

31  255  1023

Voyez vous la forme générale du résultat ?