2 Exercices sur les définitions et fonctions

Exemple 1

Définir une fonction `f` telle que $f(x) = \sqrt{(x^2+1)}/(x+1)$.
Appeler `(f 1)` retourne la valeur de $\sqrt{2}/2$.
Appeler `(f 10)` retourne la valeur de $\sqrt{101}/9$.

Solution :

```
(define (f x) (/ (sqrt (+ 1 (* x x))) (+ 1 x)))
(f 1)
(f 10)
```

```
0.7071067811865476
0.9136250564655355
```

Exemple 2

Définir une fonction `f` telle que $\begin{cases} Si\ x\ <\ 10\ alors\ f(x) = 10 \\ Sinon\ f(x) = 100 \end{cases}$
Appeler `(f 2)` retourne 10 et appeler `(f 10)` retourne 100.

Solution :

```
(define (f x) (if (< x 10) 10 100))
(f 2)
(f 10)
```

```
10
100
```

Exemple 3

Définir une fonction `f` telle que $f(x) = \sum_{i=1}^{x} i$.
Appeler `(f 2)` retourne 3 et appeler `(f 10)` retourne 55.

Première solution (récursive [25] non-terminale [26], içi récursive enveloppée [27]) :

```
(define (f x) (if (= x 0) 0 (+ x (f (- x 1)))))
(f 2)
(f 10)
```

```
3
55
```

25. La notion de récursivité est proche de la notion de définition par récurrence des suites ; à l'image de U_{n+1} définie en fonction de U_n, on aura `(f ...)` en fonction de `(f ...)` dans une définition récursive de `f`.
26. La notion de récursivité terminale est définie par une fonction dont la dernière instruction appelée est la fonction elle-même ; içi l'opérateur de multiplication `+` est appelé en dernier pour additionner `x` et `f(x-1)`.
27. La notion d'enveloppe est issue de l'addition de `f` par une valeur ; ainsi cette addition crée une enveloppe autour de `f`.

Deuxième solution (récursive terminale) :

```
(define (g x acc) (if (= x 0) acc (g (- x 1) (+ acc x))))
(define (f x) (g x 0))
(f 2)
(f 10)
```

```
3
55
```

Troisième solution (récursive terminale avec encapsulation) :

```
(define (f x)
  (define (g x acc) (if (= x 0) acc (g (- x 1) (+ acc x))))
  (g x 0))
(f 2)
(f 10)
```

```
3
55
```

Quatrième solution (récursive terminale avec `local`) :

```
(define (f x)
  (local ([define (g x acc) (if (= x 0) acc (g (- x 1) (+ acc x)))])
    (g x 0)
  ))
(f 2)
(f 10)
```

```
3
55
```

Avec l'utilisation de `local`, la portée de la définition de `g` est explicitement définie par les parenthèses.

Le développement de `(f 2)` avec la première solution empile les appels de `f`.

```
= (f 2)
= (+ 2 (f 1))
= (+ 2 (+ 1 (f 0)))
= (+ 2 (+ 1 (+ 0)))
= 3
```

Le développement de `(f 2)` avec les deuxième, troisième et quatrième solutions substitue les appels de `g` et accumule la somme.

```
= (f 2)
= (g 2 0)
= (g 1 2)
= (g 0 3)
= 3
```

Question 1

Définir une fonction `f` telle que $\begin{cases} Si\ x > 1\ alors\ f(x) = \sqrt{(x^2+1)}/(x-1) \\ Si\ x <= 1\ alors\ f(x) = 0 \end{cases}$

Appeler `(f 1)` retourne 0.
Appeler `(f 10)` retourne la valeur [28] de $\sqrt{101}/11$.
(*fonctions utiles* : `<=`, `/`, `sqrt`, `+`, `*`)

Question 2

Définir une fonction `f` telle que $f(x) = \sum_{i=1}^{x} i^2$
Appeler `(f 2)` retourne 5.
Appeler `(f 10)` retourne 385.
(*fonctions utiles* : `=`, `+`, `*`, `-`)

Question 3

Définir une fonction `f` telle que $f(x, y) = \sum_{i=x}^{y} i$.
Appeler `(f 1 2)` retourne 3.
Appeler `(f 1 10)` retourne 55.
(*fonctions utiles* : `=`, `+`, `-`)

Question 4

Définir une fonction `f` telle que $f(x, y) = \sum_{i=x}^{y} i$ avec $x > 0$ et i pair.
Appeler `(f 1 2)` retourne 2.
Appeler `(f 1 10)` retourne 30.
(*fonctions utiles* : `=`, `<`, `>`, `even?`, `+`, `-`, `modulo`)

Question 5

Définir une fonction `f` telle que $f(x, y) = x^y$ avec $y \geq 0$.
Appeler `(f 1 2)` retourne 1.
Appeler `(f 2 8)` retourne 256.
Appeler `(f 8 16)` retourne 281474976710656.
(*fonctions utiles* [29] : `=`, `*`, `-`)

Question 6

Définir la fonction factorielle `f` telle que $f(x) = \prod_{i=1}^{x} i$.
(Pour toute valeur de x inférieure ou égale à 1, $f(x)$ retourne 1).
Appeler `(f -1)` retourne 1.
Appeler `(f 1)` retourne 1.
Appeler `(f 8)` retourne 40320.

28. En réalité, une valeur approchée.
29. Pour écrire cette fonction puissance, une question introductive consiste à écrire les fonctions carré et cube dont les solutions sont içi : pour la fonction carré, on a `(define (carre x) (* x x))` ; pour la fonction cube, on a `(define (cube x) (* x (carre x)))` ou encore `(define (cube x) (* x (* x x)))` ; l'arité multiple de l'opérateur `*` qui permet également d'écrire `(define (cube x) (* x x x))` ne sera pas un indice pour l'écriture de la fonction `f` de cette question.

Appeler `(f 16)` retourne 20922789888000.
(*fonctions utiles* : `<=`, `*`, `-`)

Question 7

Définir la fonction de Fibonnacci $\mathcal{F}(x)$ telle que $\begin{cases} \mathcal{F}(0) = 0 \\ \mathcal{F}(1) = 1 \\ \mathcal{F}(n+2) = \mathcal{F}(n+1) + \mathcal{F}(n) \end{cases}$

Appeler `(F 8)` retourne 21.
Appeler `(F 16)` retourne 987.
(*fonctions utiles* : `<=`, `=`, `+`, `-`)

Question 8

Définir la fonction de Tribonnacci $\mathcal{T}(x)$ telle que

$$\begin{cases} \mathcal{T}(0) = 0 \\ \mathcal{T}(1) = 1 \\ \mathcal{T}(2) = 1 \\ \mathcal{T}(n) = \mathcal{T}(n-1) + \mathcal{T}(n-2) + \mathcal{T}(n-3) \end{cases}$$

Appeler `(T 8)` retourne 44.
Appeler `(T 16)` retourne 5768.
(*fonctions utiles* : `<=`, `=`, `+`, `-`)

Question 9

Définir une constante A égale à $\sqrt{2} + \sqrt{3}$.
(La valeur de cette constante sera retournée sous la forme d'un nombre décimal).
(*fonctions utiles* : `sqrt`, `+`)

Question 10

Définir une constante A égale à $9/5 + \sqrt{9/5}$.
(La valeur de cette constante sera retournée sous la forme d'un nombre décimal).
(*fonctions utiles* : `+`, `/`, `sqrt`)

Question 11

Définir un prédicat `est-pair?` sachant qu'être pair équivaut à avoir un modulo 2 égale à 0.
Appeler `(est-pair? 8)` retourne `#t`.
Appeler `(est-pair? 9)` retourne `#f`.
Appeler `(est-pair? 10)` retourne `#t`.
Appeler `(est-pair? 11)` retourne `#f`.
(*fonctions utiles* : `=`, `modulo`)

Question 12

Définir une fonction `pgcd` selon la méthode des différences.
Principe : un nombre est un diviseur de 2 nombres a et b s'il est un diviseur de la différence entre a et b.
Exemple pour 21 et 28 : $28 - 21 = 7$; $21 - 7 = 14$; $14 - 7 = 7$; $7 - 7 = 0$; donc 7 est pcgd de 21 et 28.
Exemple pour 8 et 13 : $13 - 8 = 5$; $8 - 5 = 3$; $5 - 3 = 2$; $3 - 2 = 1$; $2 - 1 = 1$; $1 - 1 = 0$; donc 1 est pgcd de 8 et 13.
Appeler `(pgcd 21 28)` retourne 7.
Appeler `(pgcd 8 13)` retourne 1.
(*fonctions utiles* : `=`, `<`, `-`)

Question 13

Définir une fonction `pgcd-euclidien` calculant le pgcd par la méthode des divisions.
Principe : pour deux nombres, diviser le plus grand par le plus petit ; si le reste est nul, le pgcd est le plus petit ; sinon poursuivre en divisant reste et plus petit des deux nombres.
Exemple pour 21 et 28 : $28/21 = 21 \times 1 + 7$; $21/7 = 7 \times 3 + 0$; donc 7 est le pgcd de 21 et 28.
Exemple pour 8 et 13 : $13/8 = 8 \times 1 + 5$; $8/5 = 5 \times 1 + 3$; $5/3 = 3 \times 1 + 2$; $3/2 = 2 \times 1 + 1$; $2/1 = 1 \times 1 + 1$; $1/1 = 1 \times 1 + 0$; donc 1 est le pgcd de 8 et 13.
Appeler `(pgcd-euclidien 21 28)` retourne 7.
Appeler `(pgcd-euclidien 8 13)` retourne 1.
(*fonctions utiles* : `=`, `remainder`, `<`, `-`)

Question 14

Définir une fonction `f` correspondant à $1 + 2 \times (1 + 2 \times (1 + 2 \times (\ldots)))$.
Autrement dit

$$\begin{cases} f(0) = 3 \\ f(n) = 1 + 2 \times f(n-1) \end{cases}$$

Appeler `(f 8)` retourne 1023.
Appeler `(f 16)` retourne 262143.
(*fonctions utiles* : `=`, `+`, `*`, `-`)