ISIMA 1ère année

29 janvier 2021
Durée : 2 heures
Documents autorisés

PROGRAMMATION FONCTIONNELLE

Exercice 1 :

On dispose de *n* barres de chocolat. Quand on rapporte *p* emballages au vendeur, il redonne une barre. Ecrire une fonction prenant *n* et *p* en paramètre et renvoyant le nombre maximal de barres de chocolat qui peuvent être mangées.

Exercice 2 :

1°) La fonction `SR` étant donnée, compléter la fonction `f` suivante, prenant en arguments un prédicat `P` et une liste `L`, de façon à ce qu'elle renvoie la liste des éléments de `L` vérifiant `P`.

```
(define SR
   (lambda (L B C)
      (if (null? L)
          B
          (C (car L) (SR (cdr L) B C))
       ) ))

(define f
  (lambda (P L)
    (SR L ... (lambda(....)  .... ))))
```

2°) Ecrire une ligne mettant en œuvre la fonction `f` de façon à ne conserver dans la liste des nombres `1 2 3 5 8 13 21` que les termes pairs (on pourra utiliser le prédicat prédéfini `even?` qui retourne `#t` si son argument est pair et `#f` sinon).

Exercice 3 :

Écrire une fonction `minL` prenant en entrée une liste `L`, dont les éléments sont des listes d'entiers, et qui renvoie la liste formée par les minimums de chacune des listes de L. On supposera qu'aucune liste de L n'est vide et que la fonction `min` prédéfinie prend un nombre quelconque d'arguments numériques et retourne le minimum de ces nombres.

Exemples :
`(minL '((1) (3 2)))` retourne `(1 2)`
`(minL '((4 1) (6 4 5)))` retourne `(1 4)`

Exercice 4 :

Écrire une fonction qui génère toutes les listes de *n* éléments pris dans {0,1} telles qu'il n'y ait jamais deux 1 consécutifs.

.../...

Exercice 5 :

On dispose des fonctions suivantes (**inutile de les réécrire**) :
- un prédicat `exist?` prenant en argument un prédicat unaire `P` et une liste, qui renvoie `#t` si l'un au moins des éléments de la liste vérifie `P` et `#f` sinon ;
- une fonction `iota` qui prend en entrée un entier n et qui renvoie la liste des entiers strictement plus petits que n. Par exemple `(iota 4)` retourne `(0 1 2 3)` ;
- une fonction `floor` qui prend un nombre et retourne sa partie entière.

Remarque préliminaire : $\sqrt{x} \leq \left\lfloor \frac{x}{2} \right\rfloor + 1$ (où $\left\lfloor \frac{x}{2} \right\rfloor$ désigne la partie entière de $\frac{x}{2}$).

1°) Écrire un prédicat `carre?` déterminant si un entier n est un carré
(on considérera que l'on ne dispose pas de la fonction racine carrée, ou `sqrt`).

Exemples :

- `(carre? 0) -> #t`
- `(carre? 2) -> #f`
- `(carre? 4) -> #t`
- `(carre? 1) -> #t`
- `(carre? 3) -> #f`

2°) Écrire une fonction `segment` prenant en entrée deux entiers n et p tels que $n \leq p$ et renvoyant la liste des entiers de n à p inclus.

Exemple :

- `(segment 2 5) -> (2 3 4 5)`

3°) Écrire une fonction `scn`, prenant en entrée deux entiers strictement positifs n et p, et qui renvoie `#t` si n est somme d'au plus p carrés et `#f` sinon.

Exemples :

- `(scn 4 1)` -> `#t` (car 4 = 2^2)
- `(scn 4 42)` -> `#t` (car 4 = 2^2)
- `(scn 5 1)` -> `#f` (5 n'est pas un carré)
- `(scn 5 2)` -> `#t` (5 = 2^2 + 1^2)
- `(scn 6 1)` -> `#f` (6 n'est pas un carré)
- `(scn 6 2)` -> `#f`
- `(scn 6 3)` -> `#t` (6 = 2^2 + 1^2 + 1^2)

.../...