DIU Enseigner l'informatique au lycée

Boucle sans boucle

Exercice 1 Écrire d'une manière purement fonctionnelle (sans traits impératifs) les fonctions `foo` et `bar` ci-dessous :

```
def foo(x):
    i = 0
    v = x * x
    while i <= v :
      v  = v - x
      i = i + x
    return i * v
```

```
def bar(x,y):
    r=0
    for i in range(0, 10):
      r = r + x
    for j in range(r,0,-1):
      r = r - y
    return r
```

□

Ensembles comme des fonctions

Une manière simple de représenter un ensemble d'entiers S est de définir la *fonction caractéristique* de S, nommée `car_S`, telle que `car_(x)` renvoie `True` si et seulement si `x` $\in S$. Par exemple, l'ensemble S_1 constitué de tous les entiers *strictement* positifs sera représenté par la fonction `car_S1` suivante :

```
def car_S1(x): return x > 0
```

Ainsi, pour savoir si l'entier 4 est dans S_1, il suffit d'appeler la fonction `car_S1(4)`, qui renvoie `True`, tandis que `car_S1(-3)` renvoie `False`.

Exercice 2 Donner les fonctions caractéristiques des ensembles S_2, S_3, S_4 et S_5 suivants :

- S_2 est l'ensemble de tous les entiers (positifs ou nuls) qui sont pairs ;
- S_3 est l'ensemble contenant *uniquement* les entiers 0, 3, 7 et 9 ;
- S_4 est l'ensemble des entiers entre 10 et 1000 (inclus) ;
- S_5 est l'ensemble vide.

□

Exercice 3 En utilisant cette représentation, on peut encoder les opérations habituelles sur les ensembles comme des fonctions d'ordre supérieur. Par exemple, l'opération d'union $S_1 \cup S_2$ de deux ensembles S_1 et S_2 peut être définie par la fonction `union(car_S1,car_S2)` qui renvoie la fonction caractéristique de l'union des ensembles S_1 et S_2 représentés respectivement par les fonctions `car_s1` et `car_S2`. Écrire la fonction `union` en Python. □

Exercice 4 De la même manière, donner la fonction `inter` qui réalise l'intersection de deux ensembles. □

Exercice 5 Donner la fonction `ajout` telle que `ajout(x, car_S)` renvoie la fonction caractéristique de l'ensemble dans lequel `x` a été ajouté à l'ensemble dont la fonction caractéristique est `car_S`. □

Exercice 6 En utilisant la fonction précédente, donner la fonction `ensemble` telle que `ensemble(l)` convertit une liste `l` en une fonction caractéristique de l'ensemble qui contient les valeurs de `l`. □

Récursion sans récursion

Exercice 7 Écrire une fonction `fact(n)` qui calcule la factorielle de `n` sans définir de fonctions récursives. □

Programmer avec itérateurs

En utilisant les itérateurs sur les listes `map` et `reduce`, écrire les fonctions suivantes.

Exercice 8 `somme_des_carres(l)` qui renvoie la somme des carrés des éléments d'une liste. □

Exercice 9 `maximum(l)` qui renvoie le plus grand élément d'une liste non vide. □

Exercice 10 `fst(l)` qui renvoie une liste constituée des éléments gauche d'une liste de paires. □

Exercice 11 `permute(l)` qui remplace les `0` par des `1` (et réciproquement) dans une liste `l` (constituée uniquement de `0` et de `1`). □

Exercice 12 `compte0(l)` qui compte le nombre de `0` dans la liste `l`. □

Exercice 13 `pgs(v,l)` qui renvoie la longueur de la plus grande séquence de `v` dans la liste `l`. □