I Fonctions partielles

1. On peut initialiser `m` à 0 car f est à valeur dans $\mathbb{N}$.

```
def sup(f):
    m = 0
    for k in f:
        if k > m:
            m = k
    return m
```

2. On stocke les images de f dans un dictionnaire, pour avoir le test d'appartenance en O(1) et donc une complexité totale linéaire en le nombre de clés de f :

```
def injective(f):
    images = {}
    for k in f:
        if k in images:
            return False
        images[f[k]] = True # on utilise que les clés
    return True
```

3.

```
def inverse(f):
    if not injective(f):
        return None
    g = {}
    for k in f:
        g[f[k]] = k
    return g
```

II Nombre de partitions

Soit $E_n = \{1, 2, ..., n\}$.

1. La seule partition de E_n en 1 ensemble est $\{E_n\}$, donc $p(1) = 1$.
   La seule partition de E_n en n ensembles est $\{\{1\}, \{2\}, ..., \{n\}\}$, donc $p(n) = 1$.

2. Il y a deux possibilités pour obtenir une partition $\mathcal{P}$ de E_n en k ensembles :
   - Soit n est seul dans un ensemble ($\{n\} \in \mathcal{P}$), et on doit alors partitionner E_{n-1} en $k-1$ ensembles, ce qui donne $p(n-1, k-1)$ possibilités.
   - Soit n est dans un ensemble avec d'autres éléments et il faut donc choisir une partition de E_{n-1} en k ensembles puis un ensemble auquel ajouter n, ce qui donne $k \times p(n-1, k)$ possibilités.

   Le nombre total de possibilités est donc bien $p(n, k) = p(n-1, k-1) + k \times p(n-1, k)$.

3.

```
def p(n, k):
    if k == 1 or k == n:
        return 1
    return p(n-1, k-1) + k * p(n-1, k)
```

4. `p(n, k)` est appelée effectue plusieurs fois les même sous-appels récursifs. On peut donc utiliser la programmation dynamique pour améliorer la complexité.

5.

```python
def p(n, k):
    M = [[0 for _ in range(k+1)] for _ in range(n+1)]
    for i in range(n+1):
        M[i][1] = 1
        M[i][i] = 1
    for i in range(2, n+1):
        for j in range(2, k+1):
            M[i][j] = M[i-1][j-1] + j * M[i-1][j]
    return M[n][k]
```

6. Du fait des deux boucles imbriquées, la complexité est $O(nk)$.