corrigé

Sommes de sous-ensembles

1. Préliminaires

Question 1.

```
def V(n):
    v = [26]
    for i in range(n - 1):
        v.append((353 * v[-1]) % 997)
    return v
```

2. Sommes de sous-ensembles

Question 2. On définit tout d'abord une fonction qui génère la liste S_n :

```
def sse(v, t):
    s = [0]
    for k in range(len(v)):
        si = []
        for x in s:
            if x + v[k] <= t:
                si.append(x + v[k])
        s = s + si
    return s
```

```
def somme_max(v, t):
    s = sse(v, t)
    (maxi, nb) = (0, 0)
    for x in s:
        if x > maxi:
            maxi, nb = x, 1
        elif x == maxi:
            nb += 1
    return (maxi, nb)
```

Question 3.

```
def lens(v, t):
    return len(sse(v, t))
```

Question 4. Pour tout $i \in [\![1, n]\!]$, $\operatorname{card} S_i \leqslant 2 \operatorname{card} S_i$ donc $\operatorname{card} S_i \leqslant 2^i$. Il y a égalité lorsque t est supérieur ou égal à la somme de tous les éléments de v, et dans ce cas $\operatorname{card} S_n = 2^n$. La complexité, tant temporelle que spatiale, est donc exponentielle.

Question 5.

```
def sse2(v, t):
    s = {0: 1}
    for k in range(len(v)):
        si = {}
        for x in s:
            if x + v[k] <= t:
                if x + v[k] in si:
                    si[x + v[k]] += s[x]
                else:
                    si[x + v[k]] = s[x]
        for x in si:
            if x in s:
                s[x] += si[x]
            else:
                s[x] = si[x]
    return s
```

```
def somme_max2(v, t):
    s = sse2(v, t)
    maxi = 0
    for x in s:
        if x > maxi:
            maxi = x
    return (maxi, s[maxi])
```

3. Stratégie optimale pour un jeu

Question 6.
Si on choisit de prendre l'élément v_i, l'adversaire remporte $\mathrm{M}(i+1,j)$ points et nous

$$v_i + \mathrm{S}(i+1,j) - \mathrm{M}(i+1,j) = \mathrm{S}(i,j) - \mathrm{M}(i+1,j)$$

Si on choisit de prendre l'élément v_j, l'adversaire remporte $\mathrm{M}(i,j-1)$ points et nous

$$v_j + \mathrm{S}(i,j-1) - \mathrm{M}(i,j-1) = \mathrm{S}(i,j) - \mathrm{M}(i,j-1)$$

On en déduit que $\mathrm{M}(i,j) = \max\Big(\mathrm{S}(i,j) - \mathrm{M}(i+1,j), \mathrm{S}(i,j) - \mathrm{M}(i,j+1)\Big)$

$$= \mathrm{S}(i,j) - \min\Big(\mathrm{M}(i+1,j), \mathrm{M}(i,j-1)\Big).$$

On commence par définir une fonction qui calcule les différentes valeurs $\mathrm{S}(i,j)$ pour $1 \leqslant i \leqslant j \leqslant n$:

```
def sommes(v):
    n = len(v)
    S = [[None for j in range(n)] for i in range(n)]
    for i in range(n):
        S[i][i] = v[i]
        for j in range(i + 1, n):
            S[i][j] = S[i][j - 1] + v[j]
    return S
```

puis la fonction principale, en faisant attention à l'ordre des dépendances :

```python
def gain(v):
    n = len(v)
    M = [[None for j in range(n)] for i in range(n)]
    S = sommes(v)
    for i in range(n):
        M[i][i] = v[i]
    for i in range(n - 1, -1, -1):
        for j in range(i + 1, n):
            M[i][j] = S[i][j] - min(M[i + 1][j], M[i][j - 1])
    return M[0][n - 1]
```