

1 On considère le programme suivant :

```

1  class ProgTrad {
      static int i;
3     static int k;

5     public static void main(String[] a) {
          int ic=0;
7
          while (true) {
9             switch (ic++) {
                  case 0: i=0; break;
11                case 1: k=0; break;
                  case 2: if (i>=6) ic+=3; break;
13                case 3: k+=i; break;
                  case 4: i++; break;
15                case 5: ic-=4; break;
                  case 6: System.exit(0);
17            }
          }
19     }
}

```

1. Exécuter ce programme en notant soigneusement l'évolution du contenu des variables.
2. Préciser ce que calcule ce programme.
3. Décrire son principe de fonctionnement.
4. Écrire un programme `Prog` équivalent tel qu'on l'aurait écrit idiomatiquement.
5. Écrire une variante de `ProgTrad` qui remplace l'utilisation des variables `i` et `k` par des accès à un unique tableau d'entiers `mem`.

2 La factorielle d'un entier naturel  $n$  est le produit des nombres entiers strictement positifs inférieurs ou égaux à  $n$ . On la note  $n!$  et on a donc  $n! = n \times (n - 1) \times \dots \times 1$ .

1. Écrire un programme `Factorielle` calculant itérativement la factorielle d'un nombre.
2. Traduire le programme `Factorielle` (sur le modèle de `ProgTraduit` analysé précédemment).

3 La *suite de Syracuse* d'un entier  $n$  est la suite d'entiers  $u_i$  définie par  $u_0 = n$  et, pour tout  $i \geq 0$ ,

$$u_{i+1} = \begin{cases} u_i/2 & \text{si } u_i \text{ est pair,} \\ 3 \cdot u_i + 1 & \text{sinon.} \end{cases}$$

On conjecture que toute suite de Syracuse finit toujours par atteindre le cycle (4, 2, 1). On appelle le *temps de vol* d'un entier  $n$  le plus petit indice  $i$  tel que  $u_i = 1$ .

1. Écrire un programme `Syracuse` calculant le temps de vol d'un entier  $n$ .
2. Traduire le programme `Syracuse`.

4 Traduire le programme suivant.

```
class Partiel2011{
2   public static void main(String[] arg){
        int[] t = new int[20];
4       t[0] = 1;
        t[1] = 1;
6       for(int i = 2; i < 20; i++)
            t[i] = 2 * t[i-1] + t[i-2];
8       System.out.println("Nombre(19) = " + t[19]);
        }
10 }
```