Boucles II

Le but de cette feuille est d'apprendre à concevoir la structure d'un programme sur feuille avant de se jeter sur le clavier !

Activité 1.

1. Écris un programme de compte à rebours : demande une valeur n (par exemple 10), puis affiche la liste de décompte jusqu'à 0 (soit pour l'exemple de 10 : 10, 9, 8,..., 0).
2. Écris un algorithme qui cherche le plus grand entier n tel que $n \times n \leqslant 20\,000$.
3. Trouve un algorithme qui renvoie le plus petit entier n tel que $2^n \geqslant 1\,000\,000$. (Tu peux par exemple initialiser une variable à 2 et la multiplier par 2 autant de fois que nécessaire.)
4. Écris la suite d'instructions qui teste si un nombre est premier : demande un entier n et utilise un test « est-ce que i divise n ? ». On rappelle qu'un entier n est premier s'il n'a pas de diviseurs autres que 1 et n.

Une boucle « pour... » permet de parcourir un par un tous les éléments d'une liste. Voici un exemple :

Pour i allant de 1 à 10, faire :
    afficher $i \times i$.

La variable i va prendre successivement les valeurs 1, puis 2, puis 3,... jusqu'à 10. Ce petit programme affiche les $i \times i$, c'est-à-dire dans cet ordre 1, puis 4, puis 9,... jusqu'à $10 \times 10 = 100$. La syntaxe générale est (pour a et b entiers positifs ($a < b$)) :

Pour i allant de a à b, faire :
    instruction,
    autre instruction,
    ...

L'entier i va successivement prendre la valeur a, puis $a + 1$,... jusqu'à l'entier b.

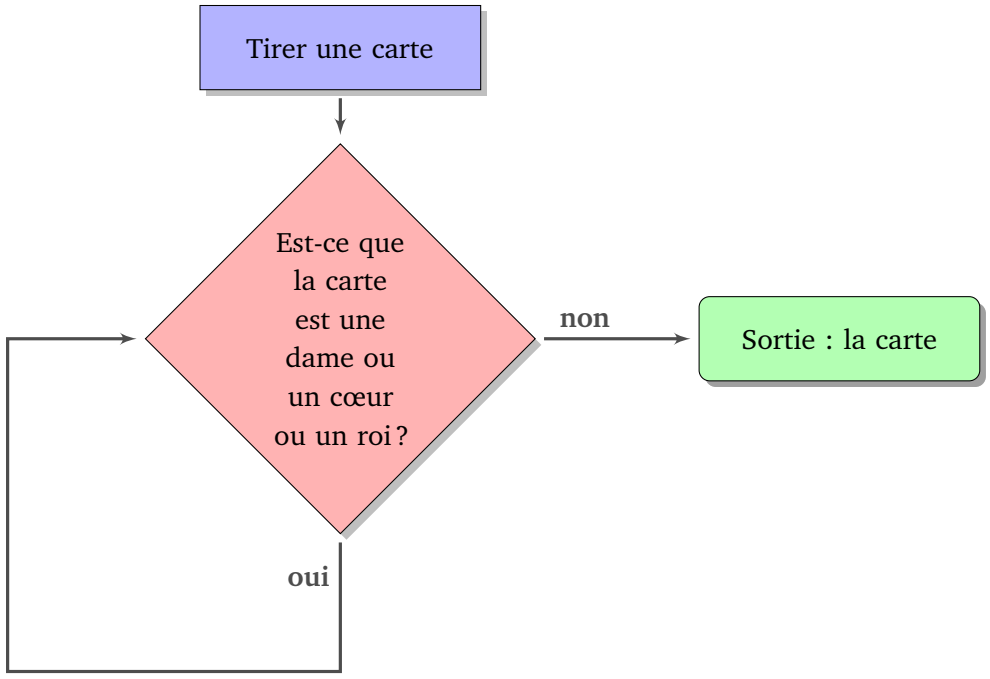
Activité 2.

1. Construis une boucle qui affiche les produits $2 \times x$, $3 \times x$,..., $20 \times x$ (où x est un nombre à demander à l'utilisateur).

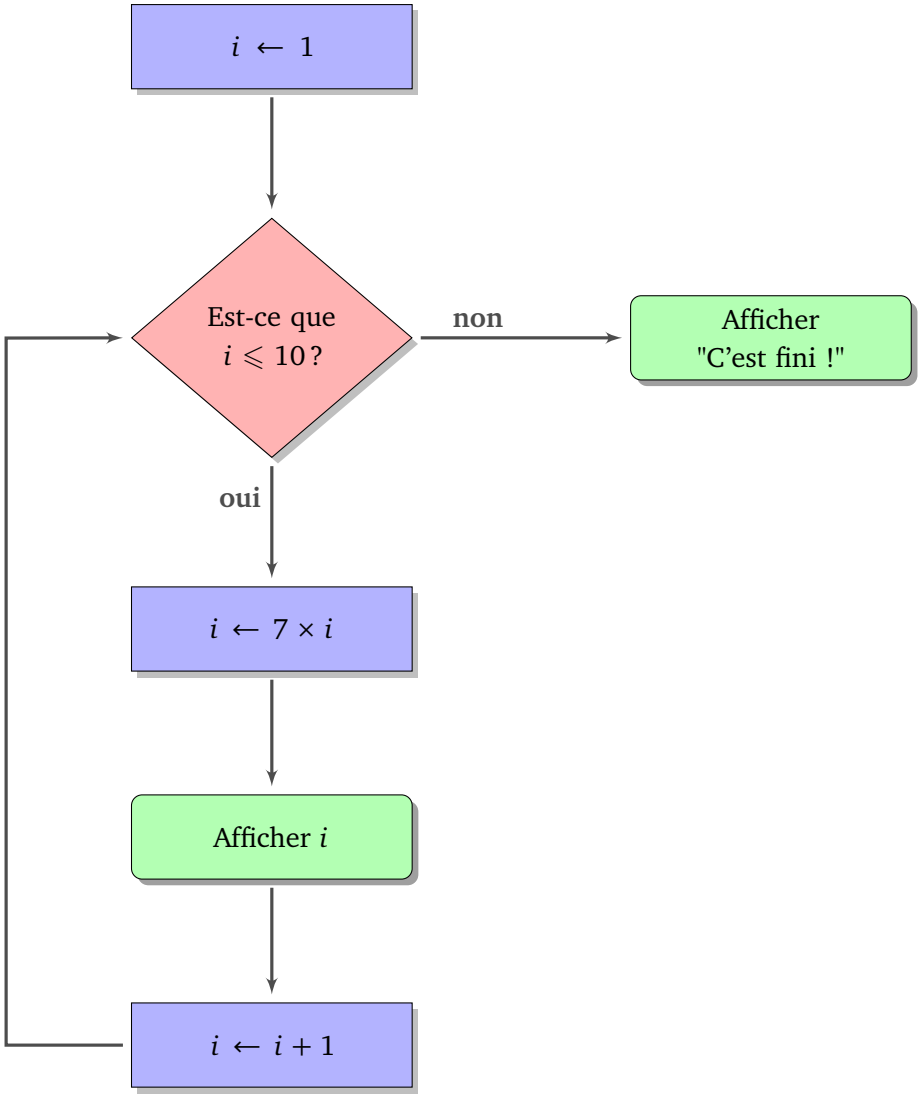
2. Construis une boucle qui calcule la somme $1 \times 1 \times 1 + 2 \times 2 \times 2 + 3 \times 3 \times 3 + \cdots + n \times n \times n$ (où n est un entier à demander à l'utilisateur).
3. Demande 10 nombres à l'utilisateur et affiche la position du plus grand de ces nombres. Par exemple, si les nombres sont $2, 3, 5, 10, 2, 1, 3, 3, 1, 5$ alors le plus grand nombre est 10 et le programme renvoie la valeur 4 (car 10 est en quatrième position).
4. Construis un programme qui affiche tous les résultats des tables classiques de multiplications (on affiche tous les produits $i \times j$, i et j étant des entiers allant de 1 à 10).

Activité 3.
Les algorithmes suivants ne font pas ce que l'on attend d'eux. Trouve les problèmes et corrige-les !

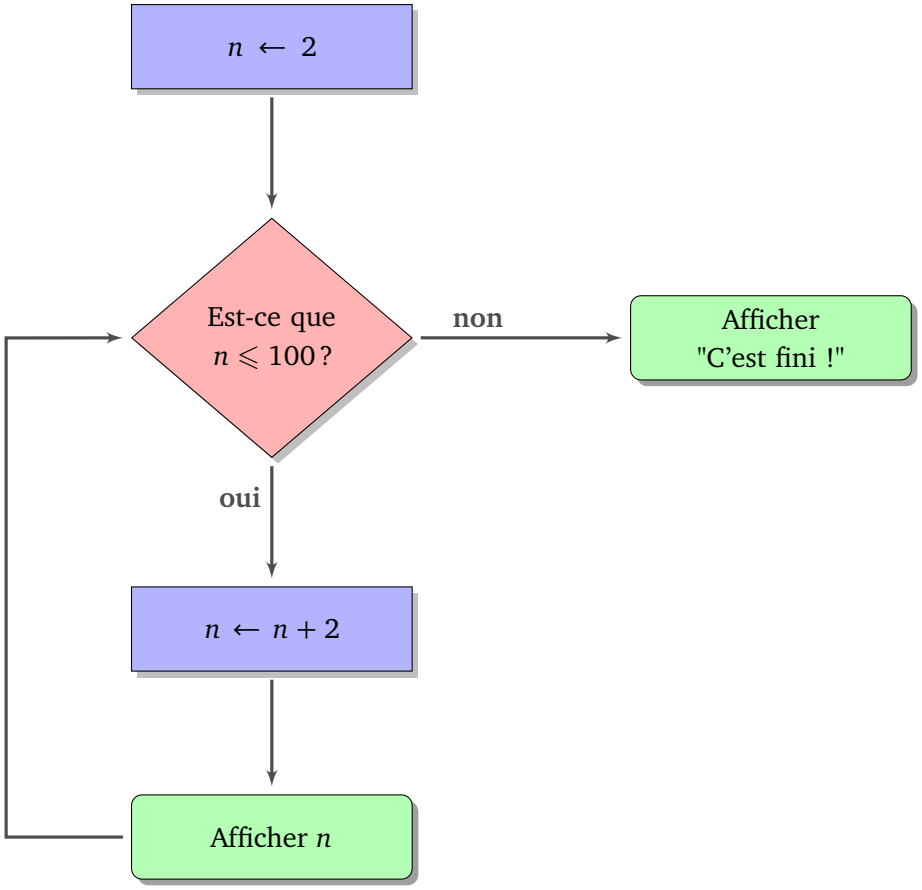
1. **But :** Le programme tire des cartes au hasard, il s'arrête lorsque la carte tirée est la dame de cœur ou le roi de cœur.
   Solution fausse :



2. **But :** Le programme affiche la table de multiplication par 7 (c'est-à-dire les multiples de 7 inférieurs ou égaux à 70).
   Solution fausse :



3. **But :** Le programme affiche les nombres pairs compris entre 2 et 100 : $2, 4, 6, 8, \ldots, 100$.
   Solution fausse :



4. **But :** Le programme calcule le produit $10 \times 9 \times 8 \times \cdots \times 2 \times 1$.
   Solution fausse :

   $P \leftarrow 1$
   $n \leftarrow 10$
   Tant que $P \geqslant 1$, faire :
   $\quad P \leftarrow P \times n$
   $\quad n \leftarrow n - 1$
   Sortie : n

5. **But :** Une somme, au départ de 1 000 €, rapporte chaque année 10 % d'intérêts (la somme d'argent est donc multipliée par 1,10 chaque année). On veut savoir au bout de combien d'années la somme placée dépasse 2 000 €.
   Solution fausse :

   $S \leftarrow 1\,000$
   $n \leftarrow 0$
   Tant que $S \geqslant 2\,000$, faire :
   $\quad S \leftarrow S \times 1,10$
   $\quad n \leftarrow n + 1$
   Sortie : $n - 1$