Bases non objet de Java

Q 1 . Déclarer et initialiser deux variables entières, puis écrire une séquence d'instructions qui échange leurs valeurs.
Q 2 . Ecrire une séquence d'instructions qui calcule le maximum de deux variables entières `x` et `y` dans une troisième variable `res`.
Q 3 . Idem avec le max de 3 nombres x, y, z, en utilisant un opérateur booléen.
Q 4 . Calculer dans `res` le PGCD de 2 entiers `x` et `y` par l'algorithme d'Euclide.
Algorithme d'Euclide :

- si un des nombres est nul, l'autre est le PGCD ;
- sinon il faut soustraire le plus petit du plus grand et laisser le plus petit inchangé ; puis, recommencer ainsi avec la nouvelle paire jusqu'à ce que un des deux nombres soit nul. Dans ce cas, l'autre nombre est le PGCD.

Q 5 . Mettre un booléen à vrai ou faux selon qu'un entier `x` est premier ou non ?
Q 6 . Initialiser un tableau `tabn` avec les entiers de 1 à `n`.
Q 7 . Somme des éléments sur la diagonale d'une matrice carrée.
Q 8 . Ranger dans `max` la plus grande valeur d'un tableau `tab`.
Q 9 . Ranger dans `index` le plus petit indice de l'élément qui vaut `valeur` dans un tableau, sinon mettre`length`.
Q 10 . Triangle de Pascal. Initialiser, pour un n donné, un tableau avec les coefficients $\mathcal{C}_n^p$, $p$ième coefficient binômial d'ordre n. Rappel :

$$\mathcal{C}_n^p = \frac{n!}{p!(n-p)!} \qquad \text{soit } \mathcal{C}_n^0 = 1, \quad \mathcal{C}_n^n = 1, \quad \mathcal{C}_n^p = \mathcal{C}_{n-1}^{p-1} + \mathcal{C}_{n-1}^p$$

À l'ordre 4 :

$n=0$	1				
$n=1$	1	1			
$n=2$	1	2	1		
$n=3$	1	3	3	1	
$n=4$	1	4	6	4	1

Pour l'ordre n on utilise un tableau `tp` de dimension 2, avec n sur la première dimension et p sur la seconde. On a donc `tp[n][p]`$= \mathcal{C}_n^p$.
Q 11 . Calculer le nombre d'entiers positifs en tête d'un tableau.
Q 12 . Calculer la taille de la plus longue séquence d'entiers positifs dans un tableau.
Q 13 . Le tri bulle. Idée de l'algorithme : parcourir les n premières cases du tableau en échangeant deux éléments successifs si le premier est plus grand que le second (soit échanger `t[i]` et `t[i+1]` si `t[i] > t[i+1]`), ce qui fait remonter comme une bulle le plus grand élément de ces n cases dans la case d'indice $n-1$, où il est bien placé. Puis on recommence en excluant du parcours les éléments bien placés. Reste à faire varier n correctement.
ex : Les étapes successives sont représentées verticalement. Après chaque étape un cadre montre le parcours de tableau restant à faire.

0	1	2	3	4	5	
4	6	5	2	1	3	tableau de départ

0	1	2	3	4	5	
4	5	2	1	3	6	après ce premier parcours l'élément d'indice 5 est maintenant bien placé.

0	1	2	3	4	5	
4	2	1	3	5	6	l'élément d'indice 4 est maintenant bien placé aussi.

0	1	2	3	4	5
2	1	3	4	5	6

0	1	2	3	4	5
1	2	3	4	5	6

0	1	2	3	4	5	
1	2	3	4	5	6	fini !