Invariant de boucle du tri par sélection

① La variable i commence à 1. $\quad A1 = A[1..i-1]$ ⚠

② $A = [27, 10, 12, 8, 11]$

$i = 1 \quad A1 = [\ ]$
ensemble vide

$A = [27, 10, 12, 8, 11]$
mini

$A = [27, 10, 12, 8, 11]$

Zone dans laquelle on cherche où est le minimum

$A = [8, 10, 12, 27, 11]$

$i = 2 \quad A1 = [8]$

mini

$A = [8, 10, 12, 27, 11]$

$A = [8, 10, 12, 27, 11]$

$i = 3 \quad A1 = [8, 10]$

mini

$A = [8, 10, 12, 27, 11]$

$A = [8, 10, 11, 27, 12]$

$i = 4 \quad A1 = [8, 10, 11]$

mini

$A = [8, 10, 11, 27, 12]$

$A = [8, 10, 11, 12, 27]$

ici, i s'arrête à 4 car A a 5 valeurs.

③ Au début de chaque itération, <u>$A1$ est trié</u>.

Montrons que "le sous-tableau $A[1..i-1]$ est trié" est un bon début pour être un invariant de boucle.

4. initialisation A l'étape 1, $i = 1$. On considère alors le sous-tableau $A[1..0]$ qui est vide. Il est donc trié.

5. conservation je suppose, au début de l'itération i, que le sous-tableau $A[1..i-1]$ est trié par ordre croissant.
Que fait le tri sélection à l'itération i ? On recherche l'indice du minimum sur le tableau $A[i..n]$ puis on procède à l'échange.
A ce moment, fin de l'étape 1B, le sous-tableau $A[1..i-1]$ est trié, et $A[i]$ est la plus petite valeur trouvée dans le reste du tableau donc $A[1..i]$ trié.
On prend maintenant en compte l'incrémentation cachée $i \to i' = i+1$.
"$A[1..i]$ trié" devient "$A[1..i'-1]$ trié" avec i' la nouvelle valeur de i au début de l'itération suivante. L'invariant est donc conservé.

6. terminaison : la boucle se termine pour $i > n-1$ soit dès que $i = n$.
A ce moment, $A[1..n-1]$ est trié. Mais $A[1..n-1]$ n'est pas tout le tableau A. Il faut, et cela manque ici, montrer que la valeur $A[n]$ est ~~la~~ bien plus grande que n'importe quelle valeur de $A[1..n-1]$.
On n'a donc pas le bon invariant de boucle, mais il ne lui manque pas grand chose. Le bon invariant de boucle est "$A[1..i-1]$ est trié ET n'importe quelle valeur de $A[i..n]$ est plus grande que $A[i-1]$".

[En réalité, on avait déjà vu que $A[i]$ étant la plus petite valeur du reste du tableau, n'importe quelle valeur restante à trier était donc plus grande que $A[i]$.

Donc, avec le bon invariant de boucle, on a $A[1..n-1]$ trié ET $A[n]$ est plus grande que n'importe quelle valeur de $A[1..n-1]$ donc $A[1..n]$ est trié, c'est à dire que le tableau A dans son entier est trié.

Cet algorithme est donc exact.

Rmq : avec le bon invariant de boucle, l'initialisation est validée car $A[1..0]$ est vide et donc il est évident que n'importe quelle valeur du reste de A est plus grande que rien.