Fonctionnement d'une boucle "Tant que .... faire"

Tant que <condition> faire Début Instructions ..... FinTantQue	Le principe de la boucle est simple : le programme arrive sur la ligne du TantQue. Il examine alors la condition. Si la condition est VRAIE, le programme exécute les instructions qui suivent, jusqu'à ce qu'il rencontre la ligne FinTantQue. Il retourne ensuite sur la ligne du TantQue, procède au même examen, et ainsi de suite. Le manège enchanté ne s'arrête que lorsque la condition est FAUSSE.

IMPORTANT : La machine "passe" et "repasse" dans la boucle tant que la condition est vraie.

Rappel du document intitulé "Module 9".

Un flou artistique réside sur l'écriture de la <condition> donc explications :

- *Une condition est composée de trois éléments :*

une valeur ***un opérateur de comparaison*** *une autre valeur*

- *Les valeurs peuvent être a priori de n'importe quel type (numériques, caractères…). Mais si l'on veut que la comparaison ait un sens, il faut que les deux valeurs de la comparaison soient du même type !*
- *Les opérateurs de comparaison sont :*
  - *égal à…*
  - *différent de…*
  - *strictement plus petit que…*
  - *strictement plus grand que…*
  - *plus petit ou égal à…*
  - *plus grand ou égal à…*

L'ensemble des trois éléments constituant la condition constitue donc, si l'on veut, une affirmation, qui a un moment donné est VRAIE ou FAUSSE.

Exemple :

Déclaration de variables
S, N et A du type nombre
Début
Affecter la valeur 5000 à S
Affecter la valeur 0 à N
Affecter la valeur 120 à A
Tant que S<6500 faire
Début
Affecter la valeur S+A à S
Affecter la valeur N+1 à N
Écrire N et S
Fin Tant que
Fin

Compléter le tableau ci-dessous des valeurs des variables au fur et à mesure de l'exécution des instructions

S	N	A

A quoi correspondent les dernières valeurs de S et N affichées ? Coder cet algorithme avec AlgoBox.

Exercice : Écrire un algorithme qui demande un nombre compris entre 10 et 20, jusqu'à ce que la réponse convienne. En cas de réponse supérieure à 20, on fera apparaître un message : « Plus petit ! », et inversement, « Plus grand ! » si le nombre est inférieur à 10.

Fonctionnement de la boucle "Pour I allant de ... à ..... "

Dans l'exemple page 1, vous avez remarqué qu'une boucle pouvait être utilisée pour augmenter la valeur d'une variable. Cette utilisation des boucles est très fréquente, et dans ce cas, il arrive très souvent qu'on ait besoin d'effectuer un nombre **déterminé** de passages. Or, a priori, notre structure TantQue ne sait pas à l'avance combien de tours de boucle elle va effectuer (puisque le nombre de tours dépend du fait que la condition est vraie ou fausse)

Déclarer I du type nombre Pour I allant de 1 à n début instructions 1 instructions 2 ..... fin	Les instructions contenues dans la boucle Pour seront exécutées n fois. (de 1 à n , il y a n entiers). A chaque passage, la valeur du "compteur" I augmente de 1. Dès que la valeur du compteur atteint n , les instructions sont exécutées une dernière fois et l'on passe à la suite de l'algorithme.

Exemple :

Déclaration de variables P, C et I du type nombre Début Affecter la valeur 1 à P Pour I allant de 1 à 10 Début Affecter la valeur I^2 à C Affecter la valeur 2P à P Écrire P et C Fin Fin	Compléter le tableau des valeurs des variables au fur et à mesure de l'exécution des instructions dans les passages successifs dans la boucle.		
	I	C	P

A quoi correspondent les valeurs de C et P affichées ? Coder cet algorithme avec AlgoBox.

Exercice : Écrire un algorithme qui demande un nombre de départ, et qui calcule sa factorielle.

Aide : factorielle 8, notée 8! , vaut $1\times2\times3\times4\times5\times6\times7\times8$.