

Les fonctions	Les procédures
<p>1. Définition : <u>Une fonction</u> est un sous programme qui renvoie une valeur de type simple (entier, booléen, caractère, chaîne de caractères). Cette valeur représente le résultat de la fonction. Exemple : sin(x)</p> <p>2. Syntaxe : <u>Au niveau de l'analyse :</u> DEF FN nom_fonction (liste des paramètres formels) : type_fonction Résultat = nom_fonction Nom_fonction ← résultat_fonction Résultat_fonction = [] {instructions} Fin nom_fonction</p> <p><u>Au niveau du Pascal :</u> Function nom_fonction (liste des paramètres formels) : type_fonction ; Var {Déclaration des objets locaux}; Begin {Instructions} ; Nom_fonction := résultat_fonction; {affectation de la valeur de retour} End ;</p> <p><u>Appel d'une fonction :</u> L'appel de la fonction doit obligatoirement apparaître dans une expression.</p> <p><u>Exemples :</u> Objet := nom_fonction (paramètres effectifs) ; Write (nom_fonction (paramètres effectifs)) ; If nom_fonction (paramètres effectifs) = ... then ... ;</p>	<p>1. Définition : Les procédures sont des sous-programmes qui peuvent avoir zéro ou plusieurs résultats.</p> <p>2. Syntaxe : <u>Au niveau de l'analyse :</u> DEF PROC nom_Procédure (liste des paramètres formels) Résultat = Instructions Fin nom_Procédure.</p> <p><u>Au niveau du Pascal :</u> Procedure nom_procédure (liste des paramètres formels); {Déclaration des objets locaux} ; Begin {Instructions} ; End ;</p> <p><u>Appel d'une procédure :</u> Pour appeler une procédure, il suffit d'écrire son nom suivi des paramètres effectifs qui se substitueront aux paramètres formels lors de l'exécution de la procédure en respectant la syntaxe suivante : Nom_procédure (paramètres effectifs) ;</p>