Documentation pour le code :

Ici on a décrit le rôle des fonctions importantes écrites dans le code ;

ALGORITHME UNE DIMENSION	
Fonction	Rôle
init_alpha(alphabet) List[str]->Dictionary	Renvoie le dictionnaire associé à la liste de str 'alphabet' ex: init_alpha(['A','T','C','G])={'A':0,'T':1, 'C':2, 'G':3}
creerTabBase(listeSeq,alphabet) List[List[str]]*Liste[str]-> List[int] or None	Initialise le Vecteur V par la liste de séquences en paramètre et l'alphabet associé
creervp (vecteurV,nbClasses) List[int]*int-> List[List[int]] or None	Créé le Vecteur P avec le Vecteur V et le nombre de classes associé. ex: dans un Vecteur V = [0,1,0,2,1,0] on a trois classes: 0,1 et 2
creervq (vp , vv ,BordSeq, k) List[List[int]]*List[int]*int ->List[List[int]] or None	Créé le Vecteur Q grâce au Vecteur P et V et un pas k. La liste d'entiers BordSeq garde les extrémités de chaque séquence pour éviter les dépassements
creervv (vq , v) List[List[int]]*List[int]->int*List[int]*List[int] or None	Créé le Vecteur V grâce au Vecteur Q et le Vecteur V précédent
ListToStr(listeSeq,k) List[str]*int->List[int]*str	Renvoie la chaîne de caractères associée à la concaténation des chaînes de la liste listeSeq
comparerSequences(listeSeq,alphabet,lenVoulue=sys.maxsize,k=1) List[str]*List[str]*int*int -> List[nuplet] or None	Renvoie les motifs qui se répètent dans listeSeq de taille maximale ou de taille lenVoulue
ALGORITHME DEUX DIMENSIONS	

Calcul(at1,at2) Atom*Atom -> Number	Calcul la distance dans l'espace entre deux atomes en utilisant leurs coordonnées
matrice(listAtom) List[Atom] -> List[List[Number]]	Sert à calculer la matrice des distance
disc_nb(nb ,pas) Number*Number ->Number	Pour discrétiser un nombre et elle renvoie la borne inférieur de l'intervalle de la taille du pas au quel le nombre appartient
disc_mat(mat) List[List[Number]]->List[List[Number]]	Calcul de la matrice discrétisée (plus facile à manipuler) à partie de la matrice des distances.
creer_vv_de_base (Mdist , dimMdist) [List[List[Number]]*int -> List[List[int]]	Créé le premier vecteur V à partir de la matrice de distances discrétisée.
creervp_2d(mat, nbSsListes)	Crée à chaque fois le nouveau vecteur P à partir du vecteur V
creervq_2d(mat,vp,k,flag)	Crée à chaque fois un nouveau vecteur Q à partir du vecteur P et du vecteur V
Creervv_dim2(vq,v,dimMdist) List[List[int]] * List[List[int] -> List[List[int]]	Créé le vecteur V, sauf le premier.
comparerSéquences_DIM2 (MDD,envoulue) List[List[int]] * int -> List[List[List[tuple(int,int)]]] * int	Trouve les séquences de forme similaires dans une protéine.
Videvq(vq) List[List[List[tuple(int,int)]] -> int	Sert a vérifier si vq est vide, et retroune le nombre de couples dans vq .
Fonctions de Lecture de fichiers	

getCAAtom (ligne) str->str*Atom ou None	Sert à détecter les carbones alpha dans les fichiers pdb.
ajouterSSEAtomes (lesAtomes, lesH, lesS) [Atom]*[int]*[int]->[Atom]	Fonction qui ajoute un atome lu dans une liste d'Atomes.
readPDB(nomFi) str->[Atom]	Est la fonction principale de lecture de fichier « .pdb » , tel que étant donné un fichier pdb renvoie la liste des carbones alpha de la protéine.
readAA(filename) str -> List [str] * int	Est la fonction de lecture de fichier «.fasta », et qui retourne la liste des séquences ainsi de la longueur totale