Variables Globales : r : Chaîne de caractères

t : Booléen

nt : Entier

Fonction md5 (→*chaine* : Chaîne de caractères) : Chaîne de caractères

Donnée(s) : *chaine* (en entrée) Le mot à transformer

Résultat : Retourne la valeur md5 du mot à transformer

Procédure reset (↔*s* : Chaîne de caractères)

Donnée(s) : *s* (en entrée/sortie) Le mot à remettre à zéro

*l* (en entrée) La lettre à mettre au début du mot

Résultat : Remet le mot en entrée à zéro avec une certaine lettre

Variable locale : *i* : Entier

Début

*s*[0] ← l

Pour *i* de 1 à taille(*s*) Faire

*s*[i] ← 'a'

FinPour

Fin

Procédure inc (↔*s* : Chaîne de caractères, ↔*v* : Chaîne de caractères, →*i* : Entier, →*l* : Caractère)

Donnée(s) : *s* (en entrée/sortie) Le mot à incrémenter

*v* (en entrée/sortie) Le mot de vérification de fin

*i* (en entrée) La position de la lettre à incrémenter

*l* (en entrée) La lettre de départ du mot incrémenter

Résultat : Incrémente le mot s pour tester la prochaine combinaison de lettre

Début

Si (*i* < 0 OU *i* > taille(*s*)) Alors

Ecrire "ERREUR : indice impossible"

Fin

FinSi

Si (*s* = *v*) Alors

reset(*s*, l)

*s* ← *s* + 'a'

*v* ← *v* + 'z'

Sinon

Si (*i* = 0) Alors

*s*[*i*]← *s*[*i*] + (*nT* - 1)

FinSi

Si (*s*[*i*] ≥ 'z') Alors

*s*[*i*] ← 'a'

inc(*s*, *v*, *i* - 1, *l*)

Sinon

*s*[*i*] ← *s*[*i*] + 1;

FinSi

FinSi

Fin

Procédure dechiffre (→*s* : Chaîne de caractères, →*v* : Chaîne de caractères)

Donnée(s) : *s* (en entrée) Le mot de combinaisons

*v* (en entrée) Le mot de vérification de fin

Résultat : Test le mot de combinaisons avec le mot à trouver jusqu'à ce qu'ils soient égaux

Variables Locales : *l* : Caractères

*dS* : Chaîne de caractères

*dR* : Chaîne de caractères

Début

*l* ← *s*[0]

*dS* ← md5(*s*)

*dR* ← md5(*r*)

TantQue (*dS* ≠ *dR* OU *t* ≠ Vrai) Faire

inc(*s*, *v*, taille(*s*) - 1, l)

*dS* ← md5(*s*)

FinTantQue

Si (*dS* = *dR*) Alors

*t* ← Vrai

FinSi

Fin