DEBUT

Donner à a la valeur 1

Donner à b la valeur 2

Donner à a la valeur a + b

Donner à b la valeur a – b

Donner à a la valeur a – b

Afficher a

Afficher b

FIN

DEBUT

Afficher « Saisissez votre note : »

Lire le nombre Note

Si Note<10 alors

Afficher « Vous n’êtes pas admis(e) »

Sinon

Si Note >= 16 alors

Afficher « Vous avez la mention Très bien ! »

Fin Si

Si Note >= 14 et Note < 16 alors

Afficher « Vous avez la mention Bien ! »

Fin Si

Si Note >= 12 et Note < 14 alors

Afficher « Vous avez la mention assez Bien ! »

Fin Si

Si Note >= 10 et Note <12 alors

Afficher « Vous êtes admis »

Fin Si

Fin Si

FIN

DEBUT

Afficher « Saisissez votre note : »

Lire le nombre note

Si note >= 16 alors

Afficher « Vous êtes admis avec la mention très bien »

Sinon

Si note >= 14 alors

Afficher « Vous êtes admis avec la mention bien »

Sinon

Si note >= 12 alors

Afficher « Vous êtes admis avec la mention assez bien »

Sinon

Si note >= 10 alors

Afficher « Vous êtes admis »

Sinon

Afficher « Vous n’êtes pas admis »

Fin Si

Fin Si

Fin Si

Fin Si

FIN

DEBUT

Afficher « Pour une équation du type ax2 + bx + c »

Afficher « Saisir A »

Lire le nombre flottant a

Si a == 0 alors

Afficher « Il ne s’agit pas d’une équation du second degré »

Stop

Fin Si

Afficher « Saisir B »

Lire le nombre flottant b

Afficher « Saisir C »

Lire le nombre flottant c

//calcul de delta

Delta <- (b \* b) – (4 \* a \* c)

STRUCTURE TVisite

IPcollaborateur : chaîne de caractères [255]

nomPage : chaîne de caractères [255]

date : chaîne de caractères [255]

navigateur : chaîne de caractères [255]

FIN

VAR

IPcumuls : tableau [1 MAXCONNECT] de TVisite

FONCTION Existe (IPTest : chaîne de caractères [255], IPcumuls tableau [1 MAXCONNECT] de TVisite , MAXCONNECT : entier)

VAR

I : entier

Retour : entier

Flag : entier

DEBUT

i <- 1

Tant que i < MAXCONNECT

SI (IPcumuls [i] . IPcollaborateur = IPTest)

Retour <- i

i <- MAXCONNECT

FIN SI

i <- i + 1

FIN Tant que

SI (flag = 0) ALORS

Retour = 0

FIN SI

Retourne retour

FIN

ALGORITHME StatistiquesSorties

CONST

MAXCONNECT = 10000

TYPE

Structure TVisite

IPcollaborateur : chaîne de caractères

NomPage : chaîne de caractères

Date : chaîne de caractères

Navigateur : chaîne de caractères

Fin Structure

...

VAR

Connexions : tableau [1..MAXCONNECT] de TVisite

...

NbreConnexions, nbreIP : entier

...

DÉBUT

NbreConnexions ← 0

i<-1

NbreIP ← 0

FichierVersTab (connexions, nbreConnexions)

TANT QUE i < MACCONNECT

Valeur<- Existe(IPcumuls, connexions[i], IPcollaborateur,MAXCONNECT)

SI valeur = 0

IPcumuls[nbConnexion +1]

adresseIP<- IPcollaborateur

IPcumuls[nbConnexion+1]

Nbre <- 1

Nbconnexion<nbConnexion+1

SINON

IPcumuls[valeur]

Nbre<-IPcumul[valeur]

Nbre+1

FIN SI

i<-i+1

FIN TANT QUE

AFFICHIER « nbconnexion total différentes » nbconnexion

FIN

FONCTION ChercheDroit(tabIds : tableau[1…1000] de chaine de caractère, tabACL : tableau[1…1000] d’ACE) : entier

VAR

I : entier

J : entier

Retour : entier

DEBUT

I <- 0

TANT QUE (I < nombreElement(tabLds))

J = 0

TANT QUE ( J < nombreElement(ACL))

SI tabIds[I] = ACL[J].UserID

SI ACL[J].UnDroit = 0

J = nombreElement(ACL)

I = nombreElement(tabLds)

Retour = 0

SINON SI ACL[J].UnDroit = 1

Retour = 1

SINON SI ACL[J].UnDroit = 2 ET Retour < 2

Retour = 2

SINON SI ACL[J].UnDroit = 3 ET Retour < 3

Retour = 3

FIN SI

SINON

Retour = 0

FIN SI

FIN TANT QUE

I++

FIN TANT QUE

RETOURNER Retour

FIN

FONCTION NombreAdresse (TabAdr : tableau d’entier)

**VAR**

Nombre : entier

Nombre2 : entier

i : entier

ChiffreDebut : entier

ChiffreFin : entier

**DEBUT**

Nombre = 0

Nombre2 = 0

**POUR** (i = 1 ; i < 508 ; i++)

**TANT QUE** TabAdr[i].AdrDeb[2] est égale a 100

**SI** TabAdr[i] .AdrDeb[1] est différent de 0 ALORS

ChiffreDebut prend la valeur : TabAdr[i] .AdrDeb[4]

ChiffreFin prend la valeur : TabAdr[i].AdrFin[4]

Nombre prend la valeur : Nombre + ChiffreFin – ChiffreDebut + 1

AFFICHER « H1 = » Nombre

**FIN SI**

i++

**FIN TANT QUE**

AFFICHER « H1 = » Nombre

**TANT QUE** TabAdr[i].AdrDeb[2] est égale à 200

**SI** TabAdr[i] .AdrDeb[1] est différent de 0 ALORS

ChiffreDebut prend la valeur : TabAdr[i] .AdrDeb[4]

ChiffreFin prend la valeur : TabAdr[i].AdrFin[4]

Nombre2 prend la valeur : Nombre2 + ChiffreFin – ChiffreDebut + 1

**FIN SI**

i++

**FIN TANT QUE**

AFFICHER « H2 = » Nombre2

**FIN POUR**

RETOURNER Nombre + Nombre2

**FIN**