**Correction de TD :**

**Exercice 1 :**

Ecrire un programme qui saisit deux entiers a et b, calcule et affiche le

quotient entier, le reste de la division et le ratio (quotient réel).

résolution :

Algorithme : division

Var a ,b,rt,qe : entier

qr : reel

DEBUT

Ecrire “entrer un nombre”

Lire a

Ecrire “entrer un nombre “

Lire b

rt ← a mod b

qe ← a div b

qr ← a / b

Ecrire “les quotient entier ”, qe

Ecrire “les quotient reel “ qr

Ecrire “le reste est “, rt

Fin

**Exercice 2 :**

Ecrire un programme qui demande à l’utilisateur de donner le rayon d’un

cercle et lui retourne sa surface et son périmètre.

PI = 4 \* arc tangente de 1. la fonction arc tangente est atan ex :

atan(2).

résolution :

Algorithme : surface

var r, s, p, pi : réel

DEBUT :

REPETER

ECRIRE (“saisir le rayon”)

LIRE ( r )

JUSQU'À (r > 0)

pi ← 4\*atan(1)

s ← pi\*r\*r

p ← 2\*pi\*r

ECRIRE (“la surface est:”, s)

ECRIRE (“le périmètre est:”, p):

Fin.

**Exercice 3 :**

Faire un programme qui saisit 3 résistances : R1, R2 et R3.

Calculer et afficher la résistance en série : R1 + R2 +R3

Calculer et afficher la résistance en parallèle : (R1 \* R2 \* R3) / (R1\*R2 +

R2\*R3 + R1\*R3)

Version 2 :

Demander a l’utilisateur d’indiquer son choix.

S’il entre la valeur 1, calculer et afficher la fréquence en série.

S’il entre la valeur 2, calculer et afficher la fréquence en parallèle.

Version 1 :

résolution :

Algorithme : Résistance

Variable :

R1, R2, R3, rest\_serie, rest\_paral, freq: reel ;

Choix : entier ;

Début

// Version 1

Ecrire (« Entrer la première résistance ») ;

Lire (R1) ;

Ecrire (« Entrer la deuxième résistance ») ;

Lire (R2) ;

Ecrire (« Entrer la troisième résistance ») ;

Lire (R3) ;

rest\_serie ← R1 + R2 +R3 ;

rest\_paral ← (R1 \* R2 \* R3) / (R1\*R2 + R2\*R3 + R1\*R3) ;

Ecrire (« La résistance en série est de :», rest\_serie) ;

Ecrire (« La résistance en parallèle est de :», rest\_paral) ;

// Version 2

Ecrire("1 - frequence en serie :" ) ;

Ecrire("2 - frequence en paralelle ") ;

Ecrire(" Faite votre choix !!! ") ;

Lire (Choix) ;

Si (Choix = 1)Alors

freq ←(R1/rest-serie) + (R2/rest-serie) + (R3/rest-serie) ;

Sinon si (Choix = 2)Alors

freq ← (R1/ rest\_paral) + (R2 / rest\_paral) +

(R3/ rest\_paral);

Sinon

Ecrire (" Vous ne pouvez choisir que 1 ou 2 ") ;

Finsi

Finsi

Ecrire ( "La fréquence est de : ", freq) ;

Fin

**Exercice 4 :**

Ecrire un programme qui saisit un réel x et un entier n et affiche x à la

puissance n. Version 1 : utiliser la fonction pow du fichier d’en-tête

<math.h> ex : pow(x,n)

Version 2 : en utilisant une boucle Version 1 :

résolution :

Algorithme : puissance

Variable : x, puissance : réel ;

n : entier ;

Début:

Ecrire (« Entrer un nombre réel ») ;

Lire (x) ;

Ecrire (« Entrer un nombre entier ») ;

Lire (n) ;

// version 1

puissance pow (x, n) ;

Ecrire (« La puissance est : », puissance) ;

// version 2

puissance ← 1

Pour (i ← 1 à n) faire

puissance ← puissance \* x ;

Finpour

Ecrire (« La puissance est : », puissance) ;

Fin

**Exercice 5 :**

Ecrire un programme qui saisit 5 variables de type entier au clavier et qui

affiche leur somme. Utiliser une boucle (for ou while ou do..while).

résolution :

Algorithme : somme

var i , somme ,val : entier

const n = 5

Debut

somme ← 0

Pour i allant de 1 a n faire

Ecrire ‘’ entrer un nombre entier ‘’

Lire val

somme ← somme + val

Finpour

Afficher ‘’ la somme des 5 valeurs est : ‘’, somme

Fin

**Exercice 6 :**

Faire un programme qui saisit les coordonnées de 2 points A (x1, y1) et

b(x2, y2) et qui affiche la distance entre les 2 points.

Formule : distante = racine carrée de ((x1 – x2)2 + (y1 – y2)2)

Racine carrée : sqrt. Ex : sqrt(7) ; <math.h>

résolution :

Algorithme: distance

Var x1, y1, x2, y2, distance: reel

Debut

Ecrire (“les coordonnées de a”)

Lire (x1, y1)

Afficher (“saisir les coordonnées de b”)

Lire (x2, y2)

distance ← sqrt (sqr(x1-x2) + sqr (y1-y2))

// distance ← sqrt (pow(x1-x2 , 2) + pow (y1-y2 , 2))

Afficher (“la distance est:”, distance)

Fin

**Exercice 7 :**

Décomposition d’un montant en euros Écrire un algorithme permettant de

décomposer un montant entré au clavier en billets de 20, 10, 5 euros et

pièces de 2, 1 euros, de façon à minimiser le nombre de billets et de

pièces.

résolution :

Algorithme: Decomposition

Var montant, billet20, billet10, billet5, piece2, piece1 ,

rest1,rest2,rest3,rest4 : reel

Debut

Repeter

Afficher “ entrer un montant “

Lire montant

Jusqu’à (montant > 0)

rest <- 0

billet20 <- ( montant div 20)

rest <- ( montant mod 20)

billet10 <- ( rest div 10)

rest <- ( rest mod 10)

billet5 <- ( rest div 5)

rest <- ( rest mod 5)

priece2 = ( rest div 2)

piece1 <- ( rest mod 2)

Ecrire “le montant’’ ,montant, ‘’est egal à ‘’, billet20

,’’billet de

20’’,, billet10 ,’’billet de 10’’, billet5 ,’’billet de 5’’, priece2

,‘’pieces de 2 et’’,priece1 ‘’pieces de 1’’

FIN

**Exercice 8 :**

Ecrire un algorithme permettant de résoudre une équation du second degré.

ax2 + bx + c = 0

résolution :

Algorithme : équation

Variable :

a, b, c, delta, x1, x2 : réel

Début

Ecrire (« Entrer les valeurs de a, b et c) ;

Lire (a, b, c) ;

Si (a = 0) alors

Ecrire (« L’équation est du premier degré ») ;

Si (b > 0) alors

Ecrire (« La solution est :», -c/b);

Sinon si (a = 0 ; b = 0 ; c = 0) alors

Ecrire (« Tout réel est solution ») ;

Sinon Ecrire (« Pas de solution») ;

Fin

si Delta b\*b – 4 ac ;

Si (delta = 0) alors

x1 = x2 = -b/2a ;

Ecrire (« On

a une racine double qui est », x1) ; Sinon si (delta > 0) alors x1 = (-b

–sqrt (delta)) / (2a); x2 = (-b –sqrt (delta)) / (2a); Ecrire («l'équation

admet 2 racines distincts réels qui sont », x1, x2) ; Sinon si (delta < 0)

alors Ecrire («l'équation n'a pas de solution dans IR») ; Fin si Fin

**Exercice 9 :**

Exercice 9 : Ecrire un algorithme qui donne la durée de vol en heure minute

connaissant l'heure de départ et l'heure d'arrivée.

a. On considère que le départ et l'arrivé ont lieu le même jour

b. On suppose que la durée de vol est inférieure à 24 heures mais peut

avoir lieu le lendemain.

resolution :

Var hh1,mn1,ss1,hh2,mn2,ss2,nbrss1,nbrss2,difference, reste, difhh,

difmn,ds : entier

Debut

REPETER

Ecrire ‘’ Entrer l’heure de depart’’

Saisir hh1,mn1,ss1

JUSQU’A (hh1 > 0 et hh1 < 24 et mn1 > 0 et mm1 < 60 et ss1 > 0 et ss1 < 60 )

REPETER

Ecrire ‘’ Entrer l’heure d’arrivé’’

Saisir hh2,mn2,s2

JUSQU’A (hh2 > 0 ET hh2 < 24 ET hh2 ET mn2 > 0 ET

mm2 < 60 ET ss2 > 0 ET ss2 < 60 )

nbrssD <- hh1 \*60 \*60 + mm1 \* 60 + ss1

nbrssA <- hh2 \*60 \*60 + mm1 \* 60 + ss1

//version a :

Si( hh2 > hh1 )

difference <- NbrssA - NbrssD

dhh <- difference div 60 \* 60

reste <- difference mod 60 \* 60

dmn <- reste div 60

dss <- reste mod 60

Ecrire ‘’ la durée du vol si le depart et l’arrivee

se tiennent à la meme journée est : ‘’ , dhh, ‘’ :’’,dmn,’’ :’’,dss

//version b

Sinon

difference <- nbrssD - nbrssA

dhh <- difference div 3600

reste <- difference mod 3600

dmn <- reste div 60

dss <- reste mod 60

Ecrire ‘’ la durée du vol si le l’arrivee

se tienne le lendemain est : ‘’ , dhh, ‘’ :’’,dmn,’’ :’’,dss

Finsi

Fin

**Exercice 10 :**

Ecrire un algorithme qui lit trois valeurs entières ( A, B et C) et qui

permet de les trier par

échanges successifs Et enfin les afficher dans l'ordre.

resolution :

Algo : trie\_par\_echange

VAR a , b , c , svg : entier ;

DEBUT :

AFFICHER ("saisir 3 nombres entiers") ;

LIRE (a,b,c) ;

SI(a > b ET b > c)Alors

svg ← 0;

FINSI

Si(a > c ET c > b) alors

svg ← b;

b ← c;

c ← svg;

FINSI

Si(b > a ET a > c) alors

svg ← a;

a ← b;

b ← svg;

FINSI

Si(b > c ET c > a) alors

svg ← b;

b ← c;

c ← a;

a ← svg ;

FINSI

Si(c > a ET a > b) alors

svg ← c;

c ← b;

b ← a;

a ← svg;

FINSI

Si(c > b ET b > a) alors

svg ← c;

c ← a;

a ← svg;

FINSI

AFFICHER ("le plus grand est :",a) ;

AFFICHER ("la moyenne est :",b) ;

AFFICHER ("le plus petit est :",c) ;

FIN

**Exercice 11 :**

Ecrire un algorithme calculatrice permettant la saisie du

premier entier (a) de l'opération ( + ou – ou \* ou / : sont des caractères)

et du deuxième entier (b) et qui affiche le résultat.

Algo : operation

Var a , b : entier;

op : caractere;

result : reel;

DEBUT:

AFFICHER ("saisir le premier nombre") ;

LIRE (a) ;

AFFICHER ("saisir le deuxieme nombre") ;

LIRE (b) ;

LIRE (op) ;

JUSQU’A( op = '-' OU op = '+' OU op = '\*' OU op = '/') ;

SI (operateur = '-' ) ALORS

result ← a - b ;

FINSI

SI (operateur = '+' ) ALORS

result ← a + b ;

FINSI

SI (operateur = '\*' ) ALORS

result ← a \* b ;

FINSI

SI (operateur = '/' ET b!=0) ALORS

result ← a / b ;

FINSI

AFFICHER (" le resultat est : ",result);

FIN

**Exercice 12 :**

Un nombre est parfait s’il est égal à la somme de ses diviseurs stricts

(différents de lui-même). Ainsi par exemple, l’entier 6 est parfait car 6 =

1 + 2 + 3. Écrire un algorithme permettant de déterminer si un entier

naturel est un nombre parfait.

resolution :

VAR nbr , i , parfait, diviseur : entier

DEBUT

REPETER

ECRIRE ‘’ Entrer un nombre ‘’

SAISIR nbr

JUSQU’A ( nbr > 0)

diviseur <- 0

POUR i allant de 1 a nbr -1 FAIRE

SI ( nbr mod i = 0) ALORS

diviseur <- diviseur + i

FINSI

FINPOUR

SI (diviseur = nbr) ALORS

ECRIRE nbr,’’ est un nombre parfait ‘’

SINON

ECRIRE nbr,’’ n’est pas un nombre parfait ‘’

FINSI

FIN

**Exercice 13 :**

Faire un programme qui saisit une date (jour, mois et année) at qui

indique si la date

est valide.

Algo : Valid\_date Var jj , mm , an : entier;

DEBUT :

AFFICHER ("Saisir le jour, le mois et l'année ") ;

LIRE (jj,mm,an) ;

SI (mm > 12 OU mm < 1 ) ALORS

AFFICHER ("invalide, les mois ne doit pas depasser 12 ni inferieur a 1");

SINON SI (mm <= 7)ALORS

SI(mm mod 2 = 0 ET jj = 31)

AFFICHER ("invalide, ce mois ne compte pas 31 jours");

SINON

SI(mm = 2 ET jj = 29 ET an mod 4 != 0) ALORS

AFFICHER ("invalide, le mois de fevrier de cette annee compte 28 jours");

SINON

SI(jj < 1 ou jj >31 ou an < 1 ou an > 2020) ALORS

AFFICHER ("jour ou annee invalide"); SINON

AFFICHER (jj,"-",mm,"-",an," est valide"); FINSI FINSI

FINSI

FINSI

SINON

SI(mm mod 2 = 1 ET jj = 31)

AFFICHER ("invalide, ce mois ne compte pas 31 jours");

SINON

SI(jj < 1 ou jj >31 OU an < 1 ) ALORS

AFFICHER ("jour ou annee invalide");

SINON

AFFICHER (jj,"-",mm,"-",an," est valide"); FINSI FINSI

FINSI FIN