**Fonctions :**

# **Exercice 1 :**

Tant que t est inférieur à 21, on appelle la fonction tableMulti() prenant en paramètre p, d et f. Cette fonction va donc afficher la table de multiplication de p, de d jusqu’à f. Ensuite, on incrémente t et on ajoute 3 à d et 5 à f.

Le résultat final des 3 variables sera donc :

t = 21

d = 35

f = 60

# **Exercice 2 :**

Les variables sont locales au programme principal.

# **Exercice 3 :**

2°)

Def table() :

n = 1

while n < 11 :

print(n\*9,end=“ ”)

n=n+1

3°)

Def table() :

n = 3

while n <= 5 :

print(n\*9,end=” ”)

n=n+1

4°) def table() :

print(8\*9,end=” ”)

**Algorithmique : Fonctions et procédures**

# **Exercice 1 :**

1°) On obtient : 22.

2°) Ob obtient 28.

# **Exercice 2 :**

1°)

def retourneMax(a, b) :

if a > b :

return a

return b

2°)

def sommeEntiers(m, n) :

k = 0

l = m

for i in range(m,n) :

k = k+l

l=l+1

return k

# **Exercice 3 :**

1°)

def echangeAB(a, b):

n = b

b = a

a = n

2°) a)

#Algo\_croissant

#Variables

a = 4 <entier>

b = 3 <entier>

#Traitement

Si b > a :

echangeAB(a,b)

FinSi

#Fin

b)

#Algo\_croissant

#Variables

a = 1 <entier>

b = 3 <entier>

c = 2  <entier>

#Traitement

Si a > b :

echangeAB(a,b)

FinSi

Si a > c :

echangeAB(a,c)

FinSi

Si a > c :

echangeAB(a,c)

FinSi

Si b > c :

echangeAB(b,c)

FinSi

#Fin

# **Exercice 4 :**

def estPremier(a) :

for i in range(2,a-1) :

if a%i != 0:

return False

return True

#Algo\_nombrePremier

#Variables

N = 6

#Traitement

Pour I allant de 1 à N-1 Faire :

si estPremier(i) :

Afficher(i)

FinSi

FinPour

#Fin

# **Exercice 5 :**

a)

def creerAleaList() :

from random import \*

alea = []

a = randint(1,20)

b = randint(1,20)

c = randint(1,20)

d = randint(1,20)

alea[0] = a

alea[1] = b

alea[2] = c

alea[3] = d

return alea

b)

def estCroissantList(alea) :

for i in range(1,len(alea)) :

if alea[i-1] > alea[i]:

return False

return True

c)

def estConsecutif(alea) :

for i in range(1,len(alea)) :

if alea[i-1]+1 != alea[i]:

return False

return True

d)

#Algo\_estDansLordre

#Variable

cpt = 0 <int>

grand = 0 <float>

petit = 0 <float>

posGrand = 0 <int>

posPetit = 0 <int>

tmpGrand = 0<float>

tmpPetit = 0 <float>

#Traitement

maListe = creerAleaListe()

Si NON estCroissantList(maListe):

Pour cpt allant de 0 à long(maListe) Faire :

Si petit > maListe[i] :

Petit <- maListe[i]

posPetit <- i

FinSi

Si grand < maListe[i] :

Grand <- maListe[i]

posGrand <- i

FinSi

FinPour

TmpPetit <- maListe[0]

maListe[0] <- petit

maListe[posPetit] <- tmpPetit

cpt <- cpt + 1

tmpGrand <- maListe[long(maListe)]

maListe[long(maListe)] <- grand

maListe[posGrand] <- tmpGrand

cpt <- cpt + 1

Si maListe[1] > maListe[2] :

Tmp <- maListe[1]

maListe[1] <- maListe[2]

maListe[2] <- tmp

cpt <- cpt +1

FinSi

FinSi

Afficher(« Il aura  fallu  », cpt, « permutations pour que la liste soit par ordre croissante »  )

Si estConsecutif(maListe):

Afficher(“ La liste est également consécutive”)

Finsi

#Fin

# **Exercice 6 :**

1°)

def capital(capEmprunt, duree, taux) :

//calcul du capital a rembourser

capRembourser = 0 <float>

for i in range(0,duree) :

capRembourser = caprembourser\*(1-taux)

capRembourser = capRembourser\*capEmprunt

//mensualité

mensualite = capRembourser / (duree\*capRembourser)

Afficher(« Le capital a rembourser est de :  », capRembourser)

Afficher(« Vos mensualité s’élèvent à : », mensualite)

2°)

def capitalReste()

**Fonction récursives :**

# **Exercice 8 :**

Pour f(10), on obtient, 10+9+8+7+6+5+4+3+2+1 = 55.