CORRIGE EXOS ALGORITHMIQUE

Partie 1
(variables, alternatives, répétitives)

(Pseudo-code)

Exercice 1.1 : inversion de 2 et 3 variables

Cas pour 2 variables

```
début
    variables X, Y, sauv : entier

    afficher("Entrer les 2 valeurs")
    saisir(X, Y)

    sauv ← X //variable de sauvagarde
    X ← Y
    Y ← sauv
    afficher(X, Y)
fin
```

cas pour 3 variables

```
début
    variables X, Y, Z, sauv : entier

    afficher("Entrer les 3 valeurs")
    saisir(X, Y, Z)

    sauv ← X
    X ← Y
    Y ← Z
    Z ← sauv

    afficher(X, Y, Z)
fin
```

Exercice 1.2 : calcul du prix TTC

```
début
    // déclaration des variables
    variables  prixHT, tauxTVA, prixTTC : réel
               nbrArt : entier

    // saisie des valeurs
    afficher("Entrez le prix hors taxes : ")
    saisir(prixHT)
    afficher("Entrez le nombre d'articles : ")
    saisir(nbrArt)
    afficher("Entrez le taux de TVA : ")
    saisir(tauxTVA)
    // traitement
    prixTTC ← nbrArt * prixHT * (1 + tauxTVA)
    // affichage du résultat
    afficher("Le prix toutes taxes est : ", prixTTC)
fin
```

Exercice 1.3 : Dire si un nombre est positif ou négatif (pas de zéro)

```
début
    variables   nbr : entier
                resultat : chaine

    afficher("Entrez un nombre : ")
    saisir(nbr)

    si (nbr >= 0) alors
       resultat ← "positif"
    sinon
       resultat ← "négatif"
    finSi

    afficher("Ce nombre est ", resultat)
fin
```

```
début
    variables nbr : entier
              resultat : chaine
    resultat  ← "positif"

    afficher("Entrez un nombre : ")
    saisir(nbr)

    si (nbr < 0) alors
       resultat ← "négatif"
    finSi

    afficher("Ce nombre est ", resultat)
fin
```

Exercice 1.4 : Dire si le produit de 2 nombres est positif ou négatif (pas de zéro)

```
début
    variables nbr1, nbr2 : entier
    (resultat : chaine) ← "négatif"

    afficher("Entrez deux nombres : ")
    saisir(nbr1)
    saisir(nbr2)

    si ((nbr1 > 0 ET nbr2 > 0) OU (nbr1 < 0 ET nbr2 < 0)) alors
       resultat ← "positif"
    finSi

    afficher("Leur produit est ", resultat)
fin
```

Exercice 1.5 : Dire si un nombre est positif ou négatif (prise en compte du zéro)

```
début
    variables   nbr : entier
                (Resultat : chaine) ← "positif"

    afficher("Entrez un nombre : ")
    saisir(nbr)

    si (nbr = 0) Alors
       resultat ← "nul"
    sinon
       si (nbr < 0) Alors
          resultat ← "négatif"
       finSi
    finSi

    afficher(Ce nombre est ", resultat)
fin
```

Exercice 1.6 : Dire si le produit de 2 nombres est positif ou négatif (avec zéro)

```
début
    variables   nbr1, nbr2 : entier
                (resultat : chaine) ← "négatif"

    afficher("Entrez deux nombres : ")
    saisir(nbr1, nbr2)

    si ((nbr1 = 0) OU (nbr2 = 0)) Alors
       resultat ← "nul"
    sinon
       Si ((nbr1 < 0 ET nbr2 < 0) OU (nbr1 > 0 ET nbr2 > 0)) Alors
          resultat ← "positif"
       finsi
    finSi
```

```
    afficher("Leur produit est ", resultat)
fin
```

Si on souhaite simplifier l'écriture de la condition lourde du « SinonSi », on peut toujours passer par des variables booléennes intermédiaires.

Exercice 1.7 : Affichage de la catégorie d'un enfant en fonction de son âge

```
début
    variables   age : entier
                categorie : chaine
    constantes (CADET : entier) ← 12, (MINIME : entier) ← 10
               (PUPILLE : entier) ← 8, (POUSSIN : entier) ← 6

    afficher("Entrez l'age de l'enfant : ")
    saisir() age

    si (age < POUSSIN) Alors
        categorie ← "sans catégorie"
    sinon
       Si (age < PUPILLE) Alors
          categorie ← "Poussin"
       sinon
          Si (age < MINIME ) Alors
             categorie ← "Pupille"
          sinon
             Si (age < cadet) Alors
                categorie ← "Minime"
             sinon
                categorie ← "Cadet"
             finsi
          finsi
       finsi
    finsi

    afficher("Sa catégorie est ", categorie)
fin
```

On peut évidemment écrire cet algorithme de différentes façons, ne serait-ce qu'en commençant par la catégorie la plus jeune.

Exercice 1.8 : Calcul de la facture des photocopies

```
début
    variables   nbrCopie : entier
                prix : réel

    afficher("Nombre de photocopies : ")
    saisir(nbrCopie)

    si (nbrCopie <= 10) alors
       prix ← nbrCopie * 0,1
    sinon
```

```
        Si (nbrCopie <= 30) alors
          prix ← 10 * 0,1 + (nbrCopie - 10) * 0,09
        sinon
          prix ← 10 * 0,1 + 20 * 0,09 + (nbrCopie - 30) * 0,08
        finSi
    finsi

    afficher("Le prix total est: ", prix)
fin
```

Exercice 1.9 : Calcul de l'impôt des habitants de Zorglub

```
début
    variables   sex, impot : caractère
                age : entier
                crit1, crit2 : booléen
    constantes (AGE_MIN_HOMME : entier) ← 20, (AGE_MIN_FEMME : entier) ← 18,
               (AGE_MAX_FEMME : entier) ← 35

    afficher("Entrez le sexe (M/F) : ")
    saisir(sex)
    afficher("Entrez l'âge: ")
    saisir(age)

    crit1 ← (sex = "M") ET (age > AGE_MIN_HOMME)
    crit2 ← (sex = "F") ET (age > AGE_MIN_FEMME) ET (age < AGE_MAX_FEMME)
    si (crit1 ou crit2) alors
        impot ← "Imposable"
    sinon
        impot ← "Non Imposable"
    finSi

    afficher("Cet habitant est ", impot)
fin
```

Exercice 1.10 : Les élections

Cet exercice, du pur point de vue algorithmique, n'est pas très méchant. En revanche, il représente dignement la catégorie des énoncés piégés. En effet, rien de plus facile que d'écrire : si le candidat a plus de 50%, il est élu, sinon s'il a plus de 12,5 %, il est au deuxième tour, sinon il est éliminé. Hé hé hé... mais il ne faut pas oublier que le candidat peut très bien avoir eu 20 % mais être tout de même éliminé, car l'un des autres a fait plus de 50 % et donc qu'il n'y a pas de deuxième tour !...
Moralité : ne jamais se jeter sur la programmation avant d'avoir soigneusement mené l'analyse du problème à traiter.

```
début
    variables  scoreA, scoreB, scoreC, scoreD : réel
               crit1, crit2, crit3, crit4 : booléen
               resultat : chaine
    constantes (ELU : entier) ← 50, (DEUX_TOUR : réel) ← 12,5

    afficher("Entrez les scores des quatre prétendants :")
    saisir(scoreA, scoreB, scoreC, scoreD)

    crit1 ← scoreA > ELU
    crit2 ← (scoreB > ELU) OU (scoreC > ELU) OU (scoreD > ELU)
    crit3 ← (scoreA >= scoreB) ET (scoreA >= scoreC) ET (scoreA >= scoreD)
```

```
    crit4 ← scoreA >= DEUX_TOUR
    si (crit1) alors
          resultat ← "élu au premier tour"
    sinon
       Si (crit2 OU Non(crit4)) alors
          resultat ← "Battu, éliminé, sorti !!!"
       sinon
          Si (crit3) alors
             resultat ← "en ballotage favorable"
          sinon
             resultat ← "en ballotage défavorable"
          finSi
       finsi
    finsi

    afficher("Le candidat 1 est ", resultat)
fin
```

Exercice 1.11 : Calcul du tarif d'assurance

Là encore, on illustre l'utilité d'une bonne analyse. On propose deux corrigés différents. Le premier suit l'énoncé pas à pas. C'est juste, mais c'est vraiment lourd. La deuxième version s'appuie sur une vraie compréhension d'une situation pas si embrouillée qu'elle n'en a l'air.
Dans les deux cas, un recours aux variables booléennes aère sérieusement l'écriture.
Donc, premier corrigé, on suit le texte de l'énoncé pas à pas :

```
début
    variables   age, perm, acc, ancien, tarif : numérique
                ageSup25, permisSup2, ancienSup1 : booléen
    constantes  (TARIF_BLEU : entier) ← 0, (TARIF_VERT: entier) ← 1,
                (TARIF_ORANGE: entier) ← 2, (TARIF_ROUGE: entier) ← 3,
                (NON_ASSURE: entier) ← 4

    afficher("Entrez l'âge: ")
    saisir(age)
    afficher("Entrez ²le nombre d'années de permis: ")
    saisir(perm)
    afficher("Entrez le nombre d'accidents: ")
    saisir(acc)
    afficher("Entrez le nombre d'années d'assurance: ")
    saisir(assur)

    ageSup25 ← age >= 25
    permisSup2 ← perm >= 2
    ancienSup1  ← ancien > 1

    si (Non(ageSup25) ET Non(permisSup2)) alors
       si (acc = 0) alors
          tarif ← TARIF_ROUGE
       sinon
          tarif ← NON_ASSURE
       finSi
    sinon
       si ((Non(ageSup25) ET permisSup2) OU (ageSup25 ET Non(permisSup2)) alors
           selon (acc)
              0 : tarif ← TARIF_ORANGE
```

```
            1 : tarif ← TARIF_ROUGE
            autres : tarif ← NON_ASSURE
        finSelon
    sinon
        selon (acc)
            0 : tarif ← TARIF_VERT
            1 : tarif ← TARIF_ORANGE
            2 : tarif ← TARIF_ROUGE
            autres : tarif ← NON_ASSURE
        finSelon
    finSi
finSi
si (ancienSup1 ET (tarif < NON_ASSURE)) alors
        tarif ← tarif - 1
finSi

selon (tarif)
    TARIF_BLEU : afficher("tarif bleu")
    TARIF_VERT : afficher("tarif vert")
    TARIF_ORANGE : afficher("tarif orange")
    TARIF_ROUGE : afficher("tarif rouge")
    autres : afficher("non assuré")
finSelon
fin
```

Vous trouvez cela compliqué ? Oh, certes oui, ça l'est ! Et d'autant plus qu'en lisant entre les lignes, on pouvait s'apercevoir que ce galimatias de tarifs recouvre en fait une logique très simple : un système à points. Et il suffit de comptabiliser les points pour que tout s'éclaire... Reprenons juste après l'affectation des trois variables booléennes `ageSup25`, `permisSup2`, et `ancienSup1`. On écrit :

```
tarif ← 1
si (Non(AgeSup25)) alors
    tarif ← tarif + 1
finSi
si (Non(permisSup2)) alors
    tarif ← tarif + 1
finSi
tarif ← tarif + acc
si (ancienSup1 ET (tarif < NON_ASSURE)) alors
    tarif ← tarif - 1
finSi
selon (tarif)
    ...
```

Cool, non ?

Exercice 1.12 : Affichage la table de multiplication

```
début
    variables nbr, i : entier

    afficher("Entrez un nombre : ")
    saisir(nbr)

    afficher("La table de multiplication de ce nombre est : ")
    pour i ← 1 a 10 faire
        afficher(nbr, " x ", i, " = ", nbr * i)
    finPour
```

```
fin
```

Exercice 1.13 : Calcul de la somme des N premiers nombres

```
début
    variables nbr, i, somme : entier

    afficher("Entrez un nombre : ")
    saisir(nbr)

    somme  ← 0
    Pour i ← 1 a nbr faire
       somme  ← somme  + i
    finPour

    afficher("La somme est : ", somme )
fin
```

Exercice 1.14 : Recherche du plus grand de 20 nombres

```
début
    variables nbr, i, plusGrand : entier

    plusGrand  ← 0
    Pour i ← 1 a 10 faire
        afficher("Entrez un nombre : ")
        saisir(nbr)
        Si ((i = 1) OU (nbr > plusGrand)) alors
             plusGrand ← nbr
        finSi
    finPour

    afficher("Le nombre le plus grand était : ", plusGrand)
fin
```

En ligne 3, on peut mettre n'importe quoi dans `plusGrand` , il suffit que cette variable soit affectée pour que le premier passage en ligne 7 ne provoque pas d'erreur.

Pour la version améliorée, cela donne :

```
début
    variables nbr, i, plusGrand , indexGrand : entier

    plusGrand  ← 0

    pour i ← 1 a 20 faire
       afficher("Entrez un nombre : ")
       saisir(nbr)
       si ((i = 1) OU (nbr > plusGrand )) alors
           plusGrand  ← nbr
           indexGrand  ← i
       finSi
    finPour

    afficher() "Le nombre le plus grand était : ", plusGrand
    afficher() "Il a été saisi en position numéro ", indexGrand
fin
```

Exercice 1.15 : Le rendu de monnaie

```
début
    variables achat, sommmeDue, paiement, rendu, nbr10E, nbr5E : entier

    achat ← 1      //init par nimporte quelle valeur pour entrer dans le tantque
    sommeDue ← 0

    // saisie des achats sortie par 0
    tantque (achat <> 0)                              //repeter
       afficher("Entrez le montant de l'achat : ")
       saisir(achat)
       sommeDue ← sommeDue + achat
    finTantque                                        //tantque(achat <>0)

    // saisie du paiement : boucle si on ne donne pas assez
    paiement ← 0
    tantque (paiement < sommeDue)
       afficher("Vous devez :", sommeDue, " euros")
       afficher("votre versement :")
       saisir(paiement)
       si (paiement < sommeDue) alors
           afficher("Vous ne donnez pas assez !")
       finSi
    finTantque

    rendu ← paiement - sommeDue
    nbr10E ← 0
    tantque (rendu >= 10)                 | Autre solution très rapide :
       nbr10E ← nbr10E + 1                |
       rendu ← rendu - 10                 | nbr10E = rendu/10)
    finTantque                            | rendu ← rendu % 10
    nbr5E ← 0                             | nbr5E = rendu/5)
    si (rendu >= 5) alors                 | rendu ← rendu % 5
       nbr5E ← 1                          |
       rendu ← rendu - 5                  | L'opérateur "/" permet de récupérer la valeur
    finSi                                 | entière de la division ; L'opérateur "%" (modulo)
                                          | permet de récupérer le reste de la division entière.

    afficher("Rendu de la monnaie :")
    afficher("Billets de 10 E : ", nbr10E)
    afficher("Billets de  5 E : ", nbr5E)
    afficher("Pièces de   1 E : ", rendu)
    fin
```