EXERCICE 1

```
//avec TANT QUE

programme sommeEntiers ;

variables
    N : entier ;
    resultat : entier = 0 ;
    i : entier = 1 ;

fonction lireEntier() : entier

    variable
        nb : entier ;

    début

        lire(nb) ;
        retourner(nb) ;

    fin


début

    écrire(« Saisissez la valeur de N : ») ;
    N := lireEntier() ;
    tantque (i <= N) faire

        resultat := resultat + i ;
        i := i+1 ;

    fintantque
    écrire (« Le résultat avec tantque est :
», resultat) ;

fin
```

```
//avec REPETER

programme sommeEntiers ;

variables
    N : entier ;
    resultat : entier = 0 ;
    i : entier = 1 ;

fonction lireEntier() : entier

    variable
        nb : entier ;

    début

        lire(nb) ;
        retourner(nb) ;

    fin


début

    écrire(« Saisissez la valeur de N : ») ;
    N := lireEntier() ;
    répéter

        resultat := resultat + i ;
        i := i+1 ;

    jusquà i > N
    écrire (« Le résultat avec répeter est :
», resultat) ;

fin
```

```
//avec POUR

programme sommeEntiers ;

variables
    N : entier ;
    resultat : entier = 0 ;
    i : entier ;

fonction lireEntier() : entier

    variable
        nb : entier ;

    début

        lire(nb) ;
        retourner(nb) ;

    fin


début

    écrire(« Saisissez la valeur de N : ») ;
    N := lireEntier() ;
    Pour i allant de 1 à N par pas de 1 faire

        resultat := resultat + i ;
        i := i+1 ;

    finpour
    écrire (« Le résultat avec pour est : », resultat) ;

fin
```

EXERCICE 2

```
//avec TANT QUE

programme factorielle ;

variables
    X : entier ;
    resultat : entier = 1 ;
    i : entier = 1 ;

fonction lireEntier() : entier

    variable
        nb : entier ;

    début

        lire(nb) ;
        retourner(nb) ;

    fin


début

    écrire(« Saisissez la valeur de X : ») ;
    X := lireEntier() ;
    tantque (i <= N) faire

        resultat := resultat * i ;
        i := i+1 ;

    fintantque
    écrire (« La factorielle de », X ,
« avec tantque est : », resultat) ;

fin
```

```
//avec REPETER

programme factorielle ;

variables
    X : entier ;
    resultat : entier = 1 ;
    i : entier = 1 ;

fonction lireEntier() : entier

    variable
        nb : entier ;

    début

        lire(nb) ;
        retourner(nb) ;

    fin


début

    écrire(« Saisissez la valeur de X : ») ;
    X := lireEntier() ;
    répéter

        resultat := resultat * i ;
        i := i+1 ;

    jusquà i > N
    écrire (« La factorielle de », X ,
« avec répeter est : », resultat) ;

fin
```

EXERCICE 3

```
programme secondDegre ;

variables
    A : entier ;
    B : entier ;
    C : entier ;
    D : réel ;  //Discriminant
    resultat1 : réel ;
    resultat2 : réel ;

fonction lireEntier() : entier

    variable
        nb : entier ;

    début

        lire(nb) ;
        retourner(nb) ;

    fin


début

    écrire(« Saisissez la valeur de A : ») ;
    A := lireEntier() ;
    si (A = 0) alors

        écrire(« Erreur : A ne peut pas être égal à 0. ») ;

    sinon

        écrire(« Saisissez la valeur de B : ») ;
        B := lireEntier() ;
        écrire(« Saisissez la valeur de C : ») ;
        C := lireEntier() ;
        écrire(« L'équation à résoudre est : », A , « x^2 + », B , « x + », C) ;
        D := (B * B) - (4 * A * C) ;
        si (D = 0) alors

            resultat1 := -(B / (2 * A)) ;
            écrire(« Le discriminant est nul, la solution de l'équation est donc »,
resultat1) ;

        sinon si (D > 0) alors

            resultat1 := (-B - racine(D)) / (2 * A);
            resultat2 := (-B + racine(D)) / (2 * A);
            écrire(« Le discriminant est positif, les solutions de l'équation sont donc »,
resultat1 , « et », resultat2) ;

        sinon

            écrire(« Le discriminant est négatif, il n'a donc pas de solution réelle à
l'équation ») ;

        finsi

    finsi

fin
```

EXERCICE 4

```
programme puissance ;

variables
    X : entier ;
    Y : entier ;
    Z : entier ;  //résultat

fonction lireEntier() : entier

    variable
        nb : entier ;

    début

        lire(nb) ;
        retourner(nb) ;

    fin

fonction puissance(entrée x : entier, entrée y : entier) : entier

    variable
        i : entier ;
        res : entier ;

    début

        i := 1 ;
        res := 1 ;

        tantque (i <= y) faire

            res := res * x ;
            i := i+1 ;

        fintantque

        retourner(res) ;

    fin

début

    écrire(« Saisissez la valeur de X : ») ;
    X := lireEntier() ;
    écrire(« Saisissez la valeur de Y : ») ;
    Y := lireEntier() ;
    Z := puissance(X, Y) ;
    écrire(X ,« ^ », Y , « = », Z) ;

fin
```

EXERCICE 5

```
programme recherche ;

variables
    X : entier ;
    tabEntiers : tableau [10] de entiers = [ -73, -8, 0, 6, 16, 31, 49, 63, 65, 100 ];
    rang : entier ;

fonction lireEntier() : entier

    variable
        nb : entier ;

    début

        lire(nb) ;
        retourner(nb) ;

    fin

fonction rechercheEntier(entrée tab : tableau, entrée x : entier) : entier

    variables
        min : entier = 1 ;
        max : entier = tab.taille ;
        mid : entier = (tab.taille / 2) ;
        res : entier  = 0 ;

    début

        tantque (res = 0) faire

            si (tab[mid] = x) alors

                res := mid ;

            sinon si (x < tab[mid]) alors

                max := mid ;
                mid := ((min + max)/2) + 1;

            sinon

                min := mid ;
                mid := ((min + max)/2) + 1;

            finsi

            si (min = max) alors

                res := -1 ;

            finsi

        fintantque
        renvoyer (res) ;

    fin

début

    écrire(« Saisissez la valeur de X : ») ;
    X := lireEntier() ;
    rang := rechercheEntier(tabEntiers, X) ;
    si (rang = -1) alors

        écrire(X, « n'est pas présent dans le tableau. ») ;

    sinon

        écrire(X, « se situe au rang », rang , « du tableau. » ;

    finsi

fin
```

EXERCICE 6

```
programme triTableau ;

constante
    n : entier ;

type
    tab : tableau [n] de entiers;

fonction lireEntier() : entier

    variable
        nb : entier ;

    début

        lire(nb) ;
        retourner(nb) ;

    fin

fonction remplirTableau(entrée tab : tableau, entrée x : entier) : tableau

    variable
        i : entier ;

    début

        pour i allant de 1 à n par pas de 1 faire

            tab[i] := nombreAleatoire( -10000, 10000 ) //on peut faire de -∞ à +∞

        finpour
        renvoyer(tab) ;

    fin

procédure triTableau(entrée tab : tableau)

    variables
        i : entier ;
        j : entier ;
        temp : entier ;
        trifini : booléen = faux ;

    début

        pour i allant de (tab.taille - 1) à 1 par pas de 1 faire

            tantque (trifini = faux) faire

                trifini := vrai ;
                pour j allant de 0 à i-1 par pas de 1 faire

                    si (tab[j+1] < tab[j]) faire

                        (T[j+1], T[j]) = (T[j], T[j+1])
                        temp := tab[j] ;
                        tab[j] := tab[j+1] ;
                        tab[j+1] := temp ;
                        trifini := faux ;

                    finsi

                finpour

            fintantque

        finpour

    fin
```

```
début

    écrire(« Saisissez la taille du tableau : ») ;
    n := lireEntier() ;
    tab := remplirTableau(tab, n) ;
    écrire(« Voici le tableau non trié : », tab) ;
    triTableau(tab) ;
    écrire(« Voici le tableau trié : », tab) ;

fin
```