ÉCOLE POLYTECHNIQUE MONTRÉAL

Le génie sans frontières

Questionnaire
examen final +corrigé

INF1005B

Sigle du cours

Identification de l'étudiant(e)		
Nom :	**Prénom :**	
Signature :	**Matricule :**	**Groupe :**

Sigle et titre du cours	*Groupe*	*Trimestre*
INF1005B – Programmation procédurale	**Tous**	**20103**
Professeur	***Local***	***Téléphone***
Martine Bellaïche	**M-3414**	**4679 / 5193**

Jour	*Date*	*Durée*	*Heures*
Mardi	**21 décembre 2010**	**2h30**	**9h30 – 12h00**

Documentation	*Calculatrice*	
☒ Aucune	☒ Aucune	**Les cellulaires, agendas électroniques ou téléavertisseurs sont interdits.**
☐ Toute	☐ Toutes	
☒ Voir directives particulières	☐ Non programmable	

Directives particulières

- **Ne recopiez pas les déclarations ni les instructions déjà fournies dans le questionnaire.**
- **L'option Explicit est à ON.**
- **L'option Strict est à ON.**
- **En annexe, vous avez la liste des fonctions de Visual Basic.**
- **Vous n'avez pas à écrire de commentaires ni d'en-têtes.**
- **En cas de doute, veuillez faire vos suppositions et les écrire sur le cahier d'examen.**

Bon succès à tous!

Important

Cet examen contient **5** questions sur un total de **13** pages **(excluant cette page)**

La pondération de cet examen est de **40** **%**

Vous devez répondre sur : ☐ le questionnaire ☒ le cahier ☐ les deux

Vous devez remettre le questionnaire : ☐ oui ☒ non

L'étudiant doit honorer l'engagement pris lors de la signature du code de conduite.

Question 1 (2.5 points)

Répondre par Vrai ou faux.

1.1 Si dans l'environnement Visual Studio, l'option Explicit est à ON, alors toutes les variables doivent être déclarées.

1.2 Si dans l'environnement de Visual Studio, l'option Strict est à ON, alors ces deux instructions compilent.

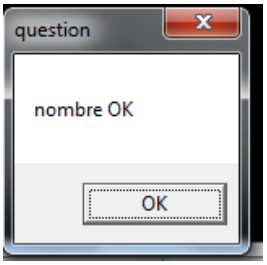
```
Dim intAge As Integer
intAge = InputBox("Veuillez donner votre age")
```

1.3 Le corps de la boucle `Do While condition … Loop` est exécuté si et seulement si la condition est fausse.

1.4 Le corps de la boucle `Do … loop until condition` est exécuté si et seulement si la condition est fausse.

1.5 Si l'usager rentre la valeur 12.789, après l'exécution de ces instructions, le programme affichera la fenêtre suivante.

```
Dim dblNombre As String
 dblNombre = InputBox("donner un nombre")
 If dblNombre Like "#.#" Then
     MsgBox("nombre OK")
 Else
     MsgBox("nombre non OK")
 End If
```



Réponses question 1 0.5 par réponse exacte

1. Vrai
2. Faux
3. Faux
4. Vrai
5. faux

Question 2 (6 points)

2.1 À l'aide de l'instruction de répétition `for`, écrire les déclarations et les instructions qui font la somme des nombres impairs entre 1 et 50 inclusivement. La somme est affichée par la suite.

2.2 À l'aide de l'instruction de répétition `Do…. Loop Until`, écrire les déclarations et les instructions qui répètent la saisie d'un nombre devant être compris entre 0 et 100 inclusivement.

2.3 Écrire la définition des structures permettant de représenter un avion : le nom du fabricant (Boeing, Airbus, Bombardier, Embraer, …), le type de l'avion : 777-300ER, A330-300, …, le nombre de places en classe économique, le nombre de places en classe affaire et la vitesse de croisière. De plus, pour un avion, on désire aussi conserver les coordonnées du fabricant (adresse et téléphone).

Question 2 (suite)

2.4 Écrire **seulement l'entête** d'une sub-routine qui trouve la valeur minimale, la valeur maximale d'un tableau d'entiers à deux dimensions. La valeur minimale, la valeur maximale et le tableau devraient être des paramètres.

2.5 Écrire **seulement l'entête** d'une fonction. Ayant le nom d'un fichier, la fonction doit vérifier l'existence du fichier et trouver le nombre de lignes. Si le fichier n'existe pas, on obtient la valeur fausse, si le fichier existe, on a la valeur vraie et la fonction calcule le nombre de lignes du fichier. Le nom du fichier, la vérification du fichier, le nombre de lignes devraient être des paramètres ou une valeur de retour de la fonction.

Solution question 2

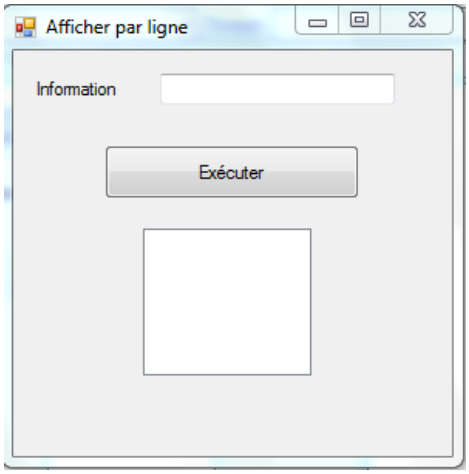
```
Sub Main()
        'reponse  2.1
        Dim somme As Integer = 0
        For i As Integer = 1 To 50 Step 2
            somme = somme + i
        Next
        MsgBox(" somme " & somme)
        'réponse 2.2
        Dim nombre As Integer
        Do
            nombre = CInt(InputBox(" donner un nombre "))
        Loop Until (nombre >= 0) And (nombre <= 100)
    End Sub
    'réponse 2.3
    Structure adresse
        Dim intNumero As Integer
        Dim strRue As String
        Dim strCodePostal As String
        Dim strVille As String
        Dim strPays As String
        Dim intTelephone As Integer
    End Structure

    Structure avion
        Dim strFabriquant As String
        Dim strAdresse As adresse
        Dim strTypeAvion As String
        Dim intPlaceEconmique As Integer
        Dim intPlaceAffaire As Integer
        Dim sngVitesseCroisiere As Single

    End Structure
    'reponse  2.4
    Sub trouverMinmax(ByVal tableau2d(,) As Integer, _
                      ByRef min As Integer, _
                      ByRef max As Integer)
    End Sub
    'reponse 2.5
    Function CalculerLignesFichier(ByVal nomFichier As String, _
                                   ByRef nombreLignes As Integer) _
                                   As Boolean
    End Function
```

Question 3 (2.5 points)

Soit une application window `AfficherParLigne` comprenant les toolbox suivants : `information` de type label, `textInformation` de type textBox, `executer` de type button, `caractereParLigne` de type listBox.



Dans la classe `Form1`, on désire afficher un caractère par ligne dans le toolbox `listBox` de l'information donnée par l'usager dans le toolbox `textBox`. L'information est une phrase. Seulement les lettres ou les chiffres seront affichés. Écrire les déclarations et les instructions manquantes dans le programme ci –dessous.

```
Public Class Form1

    Private Sub executer_Click(ByVal sender As System.Object, ByVal e As  _
System.EventArgs) Handles executer.Click


    End Sub

    Private Sub TexteInformation_TextChanged(ByVal sender As System.Object, _
ByVal e As System.EventArgs) Handles TexteInformation.TextChanged

    End Sub

End Class
```

Réponse question 3

```
Dim strInformation As String

Private Sub executer_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles executer.Click
        For i As Integer = 0 To strInformation.Length() - 1
            If Char.IsLetterOrDigit(strInformation(i)) Then
                CaractereParLigne.Items.Add(strInformation(i))
            End If
        Next

 End Sub

Private Sub TexteInformation_TextChanged(ByVal sender As System.Object, ByVal
e As System.EventArgs) Handles TexteInformation.TextChanged
        strInformation = TexteInformation.Text
End Sub
```

Question 4 (6 points)

On vous demande d'écrire un programme en Visual Basic permettant de corriger les réponses des étudiants à des questions vraies ou fausses et de trouver le nombre de réponses exactes. Pour résoudre ce problème, on a deux fichiers :

- un fichier contenant les questions et les réponses exactes. Sur deux lignes, on retrouve la question et ensuite la réponse à la question. On ne connaît pas le nombre de questions contenues dans le fichier. Voici un exemple de fichier :

```
Le symbole% n'est pas un operateur de visual basic
vrai
La fonctiom membre String.Format permet de controler le formatage du texte
vrai
end dans une fonction retourne une valeur lors de l'appel de la fonction
faux
La methode readline de streamReader lit une ligne a la fois dans un fichier
vrai
```

……..

- un fichier contenant seulement le nom de l'étudiant et les réponses de celui-ci. Les réponses sont dans le même ordre que les questions du fichier questions et réponses. On suppose que l'étudiant a répondu à toutes les questions. Voici un exemple de fichier de réponses d'un étudiant :

```
Alain Tremblay
vrai
faux
faux
vrai
```

…….

En utilisant seulement les sous-programmes `InputBox` et `MsgBox`, le programme devra :

1. Demander les noms des fichiers à l'usager et vérifier leur existence.
2. Lire les deux fichiers sachant que l'on ne connaît pas le nombre de questions.
3. Afficher les questions dont la réponse de l'étudiant est exacte.
4. Pour chaque bonne réponse de l'étudiant, ajouter un point à l'étudiant.
5. Une fois terminer la lecture des fichiers, afficher le nom de l'étudiant et le total de ses points.

Notes : Afficher les messages appropriés pour chaque sous-question.

Réponse question 4

```
Sub Main()
        Dim strNomEtudiant As String
        Dim strQuestion As String
        Dim strReponse As String
        Dim strReponseEtudiant As String
        Dim intNombrePoints As Integer = 0
        Dim strFichierQuestionReponse As String
        Dim strFichierReponseEtudiant As String
        Dim fichierQuestionReponse As IO.StreamReader
        Dim fichierReponseEtudiant As IO.StreamReader

        strFichierQuestionReponse = InputBox("Donner le nom du fichier de
Question et Réponse")
        strFichierReponseEtudiant = InputBox("Donne le nom du fichier des
réponses de l'étudiant")
        If (IO.File.Exists(strFichierQuestionReponse) And _
            (IO.File.Exists(strFichierReponseEtudiant))) Then
            'fichierQuestionReponse =
IO.File.OpenText(strFichierQuestionReponse)
            'fichierReponseEtudiant =
IO.File.OpenText(strFichierReponseEtudiant)
            fichierQuestionReponse = New
IO.StreamReader(strFichierQuestionReponse)
            fichierReponseEtudiant = New
IO.StreamReader(strFichierReponseEtudiant)
            strNomEtudiant = fichierReponseEtudiant.ReadLine()
            Do While Not fichierQuestionReponse.EndOfStream()
                strQuestion = fichierQuestionReponse.ReadLine()
                strReponse = fichierQuestionReponse.ReadLine()
                strReponseEtudiant = fichierReponseEtudiant.ReadLine()
                If (strReponse = strReponseEtudiant) Then
                    MsgBox(strQuestion & " est exacte")
                    intNombrePoints += 1
                Else
                    MsgBox(strQuestion & " n'est pas exacte")
                End If
            Loop
            MsgBox(strNomEtudiant & " a obtenu " & intNombrePoints)

        Else

            MsgBox("Les fichiers n'existent pas ")

        End If
    End Sub
```

Question 5 (3 points)

Pour le temps des fêtes, et ayant maintenant une bonne connaissance du langage Visual Basic, vous décidez d'écrire un programme permettant de gérer votre liste de cadeaux de Noël. Dans un fichier binaire, on y retrouve la liste de vos amis. Pour chaque ami, on conserve le cadeau que vous lui avez acheté, ainsi que son prix selon la structure suivante.

```
Structure CadeauInfo
        Dim strNom As String
        Dim strNomCadeau As String
        Dim dblPrix As Double
End Structure
```

On suppose que le fichier binaire est déjà créé. En vous aidant des déclarations et une partie des instructions de la page suivante, veuillez écrire les déclations et les instructions pour réaliser les opérations suivantes :

1. Déclaration des variables pour lire un fichier binaire sachant que le nom du fichier Window est une variable de type string.

2. Calculer le nombre de structures de type `CadeauInfo` dans le fichier binaire. Vous pouvez vous servir des déclarations existantes.

3. Selon le nom d'un ami, trouver dans le fichier binaire les informations correspondantes et afficher le cadeau correspondant à cet ami. La recherche doit se terminer, si l'on a affiché le nom du cadeau de l'ami. Si la recherche a été non fructeuse, on affichera aussi un message signalant que l'on n'a pas trouvé d'informations sur cet ami. Par la suite, on fermera les fichiers.

Note : Pour l'affichage, on utilisera MsgBox().

Question 5 (suite)

```
'votre réponse à la question 5.1

'déclarer une variable permettant de conserver le nom du fichier
' cadeau.bin
'déclarer les variables pour lire un fichier binaire

' variable pour la recherche d'un ami
Dim trouve As Boolean = False
' variable pour compter le nombre d'ami dans le fichier binaire
Dim sInfotemp As CadeauInfo
' initialisation pour permettre de calculer le nombre d'octets
' d'une structure
sInfotemp.strNom = "inconnu"
sInfotemp.strNomCadeau = "rien"
sInfotemp.dblPrix = 0.0
' compteur
 Dim compteur As Integer = 0
'nombre d'amis dans le fichier binaire
Dim intNombreAmi As Integer
' nom d'un ami
Dim strNomAmi As String


' Votre réponse à la question 5.2
' calcul du nombre de structures ou du nombre d'ami dans le
'fichier binaire

Console.WriteLine("Il y a " & intNombreAmi & " entrées dans le fichier " & _
                  strNomFichier & Environment.NewLine)



' Votre réponse à  la question 5.3

'Demander à l'usager le nom de votre ami
strNomAmi = InputBox("quel est le nom de votre ami")

' se placer au  début  du fichier binaire
' boucle pour lire dans le fichier binaire
' afin d'afficher le nom du cadeau de l'ami
' de  la  variable strNomAmi
' on termine la boucle si on a trouvé l'ami
' afficher un message si on n'a pas trouvé  l'ami
'fermeture des fichiers
```

```
Réponse question 5
'déclarer une variable permettant de conserver  le nom du fichier
        ' cadeau.bin
        'déclarer les variables pour lire un fichier binaire
        Dim fichier As FileStream
        fichier = File.OpenRead(strNomFichier)
        Dim binReader As New BinaryReader(fichier)
        ' variable pour la recherche d'un ami
        Dim trouve As Boolean = False
        ' variable pour  compter le nombre d'ami dans le fichier binaire
        Dim sInfotemp As CadeauInfo
        ' initialisation pour permettre de calculer le nombre d'octets
        ' d'une structure
        sInfotemp.strNom = "inconnu"
        sInfotemp.strNomCadeau = "rien"
        sInfotemp.dblPrix = 0.0
        ' compteur
        Dim compteur As Integer = 0
        'nombre d'amis dans le fichier binaire
        Dim intNombreAmi As Integer
        ' nom d'un ami
        Dim strNomAmi As String
        ' calcul du nombre de structures ou du nombre d'ami dans le
        'fichier binaire
        intNombreAmi = CInt(fichier.Seek(0, SeekOrigin.End))
        intNombreAmi = CInt(intNombreAmi / _
         (Len(sInfotemp.strNom)+ Len(sInfotemp.strNomCadeau) +
Len(sInfotemp.dblPrix)))

        Console.WriteLine("Il y a " & intNombreAmi & " entrées dans le
fichier " & _
                          strNomFichier & Environment.NewLine)

        'Demander à l'usager le nom de votre ami
        strNomAmi = InputBox("quel est le nom de votre ami")

        ' se placer au  début  du fichier binaire
        fichier.Seek(0, SeekOrigin.Begin)
        ' boucle pour lire dans le fichier binaire
        ' afin d'afficher le nom du cadeau de l'ami
        ' de  la  variable strNomAmi
        ' on termine la boucle si on a trouvé l'ami
        Do While Not trouve And compteur < intNombreAmi
            sInfotemp.strNom = binReader.ReadString
            sInfotemp.strNomCadeau = binReader.ReadString
            sInfotemp.dblPrix = binReader.ReadDouble
            If (sInfotemp.strNom = strNomAmi) Then
                MsgBox(sInfotemp.strNom & " a le cadeau " &
sInfotemp.strNomCadeau)
                trouve = True
            End If
            compteur += 1
        Loop
        ' afiicher un message si on n'a pas trouvé  l'ami
        If Not trouve Then
            MsgBox(" cet ami n'est pas dans votre liste de cadeau")
        End If
```

```
        'fermeture des fichiers
        binReader.Close()
        fichier.Close()
```

Annexe

Boolean, Byte, Short Integer. Long, Single, Double, Decimal Char, String, Date	
IsNothing(ByVal *Expression* As type) As Boolean	Vérifie si l'expresion possède une valeur
IsNumeric(ByVal *Expression* As type) As Boolean	Vérifie si l'expression peut être convertie en valeur numérique
Environment.NewLine	Changement de ligne
.GetType().ToString()	Retourne le type de la variable et ensuite est converti en string
Byte, Short, Integer, Char Constantes	
Byte.MaxValue Short.MaxValue Integer.MaxValue Char.MaxValue	Plus grande valeur
Byte.MinValue Short.MinValue Integer.MinValue Char.MinValue	Plus petite valeur
Single Double	
Constantes	
Single.Epsilon Double.Epsilon	La plus petite valeur réelle après le 0
Single.MaxValue Double.MaxValue	La plus grande valeur
Single.MinValue Double.MinValue	La plus petite valeur
Single.NaN Double.NaN	Résultat indéfini tel que la division de 0 par 0
Single.NegativeInfinity Double.NegativeInfinity	Division d'un nombre négatif par zéro ou résultat trop petit.
Single.PositiveInfinity Double.PositiveInfinity	Division d'un nombre positif par zéro ou résultat trop grand.
Fonctions Membres	
.IsInfinity(ByVal *f* As Single or Double) As Boolean	Si f est égale à NegativeInfinity ou PositiveInfinity.
.IsNaN(ByVal *f* As Single or double) As Boolean	Si f est égale à NaN.
.IsNegativeInfinity(ByVal *f* As Single or double) As Boolean	Si f est égale à NegativeInfinity.
.IsPositiveInfinity(ByVal *f* As Single or double) As Boolean	si f est égale à PositiveInfinity.
Decimal	
Constantes	
Decimal.MaxValue	La plus grande valeur
Decimal.MinValue	La plus petite valeur
Char	
.IsDigit(ByVal *c* As Char) As Boolean	Vérifie si le caractère est un chiffre (0-9)
.IsDigit(ByVal *s* As String, ByVal *index* As Integer) As Boolean	Vérifie si le caractère de s à la position index est un chiffre (0-9)
.IsLetter(ByVal *c* As Char) As Boolean	Vérifie si le caractère est une lettre (a-z)
.IsLetter(ByVal *s* As String, ByVal *index* As Integer) As Boolean	Vérifie si le caractère de s à la position index est une lettre (a-z)
.IsLetterOrDigit(ByVal *c* As Char) As Boolean	Vérifie si le caractère est une lettre ou un chiffre
.IsLetterOrDigit(ByVal *s* As String, ByVal *index* As Integer) As Boolean	Vérifie si le caractère de s à la position index est une lettre ou un chiffre

.IsLower(ByVal *c* As Char) As Boolean	Vérifie si le caractère est une lettre minuscule
.IsLower(ByVal *s* As String, ByVal *index* As Integer) As Boolean	Vérifie si le caractère de s à la position index est une lettre minuscule.
.IsPunctuation(ByVal *c* As Char) As Boolean	Vérifie si le caractère est une ponctuation
.IsPunctuation(ByVal *s* As String, ByVal *index* As Integer) As Boolean	Vérifie si le caractère de s à la position index est une ponctuation
.IsUpper(ByVal *c* As Char) As Boolean	Vérifie si le caractère est une lettre majuscule
.IsUpper(ByVal *s* As String, ByVal *index* As Integer) As Boolean	Vérifie si le caractère de s à la position index est une lettre majuscule.
.IsWhiteSpace(ByVal *c* As Char) As Boolean	Vérifie si le caractère est un espace
.IsWhiteSpace(ByVal *s* As String, ByVal *index* As Integer) As Boolean	Vérifie si le caractère de s à la position index est un espace
.ToLower(ByVal *c* As Char) As Char	Convertit le caractère en minuscule
.ToUpper(ByVal *c* As Char) As Char	Convertit le caractère en majuscule
.ToString(ByVal *c* As Char) As String	Convertit en String
AscW(ByVal c as Char) as Integer	Convertir char en integer
ChrW(ByVal I as Integer) as char	Convertir integer en char
Chr(ByVal I as Integer) as char	Retourne le caractère de la position i de la table Ascii.
Asc(ByVal c as char) as Integer	Retourne la valeur numérique de c de la table Ascii.
String	
Opérateur s (=, <, >, <=, >=)	Comparaison
String.Concat(ByVal *str0* As String, ByVal *str1* As String) As String	Concaténer deux chaînes (idem avec l'opérateur &)
String.Concat(ByVal *str0* As String, ByVal *str1* As String, ByVal *str2* As String) As String	Concaténer trois chaînes (idem avec l'opérateur &)
.Contains(ByVal *value* As String) As Boolean	Vérifie si une chaîne value apparaît dans la variable.
.EndsWith(ByVal *value* As String) As Boolean	Vérifie si la chaîne value apparaît à la fin de la variable.
.IndexOf(ByVal *value* As Char) As Integer	Recherche la position de la première occurrence du caractère ou string value dans la variable. -1 non trouvé.
.IndexOf(ByVal *value* As Char, ByVal *startIndex* As Integer) As Integer .IndexOf(ByVal *value* As String, ByVal *startIndex* As Integer) As Integer	Recherche la position de la première occurrence du caractère ou string value dans la variable à partir de la position startIndex. -1 non trouvé.
.IndexOf(ByVal *value* As Char, ByVal *startIndex* As Integer, ByVal *count* As Integer) As Integer .IndexOf(ByVal *value* As String, ByVal *startIndex* As Integer, ByVal *count* As Integer) As Integer	Recherche la position de la première occurrence du caractère ou string value dans la variable à partir de la position startIndex et pour count caractères. -1 non trouvé.
.IndexOfAny(ByVal *anyOf*() As Char) As Integer	Recherche la position de la première occurrence d'un des caractères contenus dans anyof() dans la variable. -1 non trouvé
.IndexOfAny(ByVal *anyOf*() As Char, ByVal *startIndex* As Integer) As Integer	Recherche la position de la première occurrence d'un des caractères contenus dans anyof() dans la variable à partir de la position startIndex. -1 non trouvé.

.IndexOfAny(ByVal *anyOf*() As Char, ByVal *startIndex* As Integer, ByVal *count* As Integer) As Integer	Recherche la position de la première occurrence d'un des caractères contenus dans anyof() dans la variable à partir de la position startIndex et pour count caractères. -1 non trouvé.
.Insert(ByVal *startIndex* As Integer, ByVal *value* As String) As String	Permet d'insérer une chaîne à l'intérieur de la variable à la position spécifiée
String.Join(ByVal *separator* As String, ByVal *value*() As String) As String	Concatène le tableau value() de string séparé par separator.

.LastIndexOf(ByVal *value* As Char) As Integer	Recherche la position de la dernière occurrence du caractère value. -1 non trouvé.
.LastIndexOf(ByVal *value* As Char, ByVal *startIndex* As Integer) As Integer	Recherche la position de la dernière occurrence du caractère value à partir de la position startIndex de la variable. -1 non trouvé.
.LastIndexOf(ByVal *value* As Char, ByVal *startIndex* As Integer, ByVal *count* As Integer) As Integer	Recherche la position de la dernière occurrence du caractère value à partir de la position startIndex de la variable et pour count caractères. -1 non trouvé
.LastIndexOfAny(ByVal *anyOf*() As Char) As Integer .LastIndexOfAny(ByVal *anyOf*() As Char, ByVal *startIndex* As Integer) As Integer .LastIndexOfAny(ByVal *anyOf*() As Char, ByVal *startIndex* As Integer, ByVal *count* As Integer) As Integer	Idem mais pour une recherche à partir d'un tableau de caractères.
.PadLeft(ByVal *totalWidth* As Integer) As String .PadLeft(ByVal *totalWidth* As Integer, ByVal *paddingChar* As Char) As String .PadRight(ByVal *totalWidth* As Integer) As String .PadRight(ByVal *totalWidth* As Integer, ByVal *paddingChar* As Char) As String	Placer des espaces ou un caractère spécifique à gauche ou droite pour *totalWidth* de la variable.
.Remove(ByVal *startIndex* As Integer) As String	Supprimer tous les caractères de la variable à partir de *startIndex*.
.Remove(ByVal *startIndex* As Integer, ByVal *count* As Integer) As String	Supprimer *count* caractères de la variable à partir de *startIndex*.
.Replace(ByVal *oldChar* As Char, ByVal *newChar* As Char) As String .Replace(ByVal *oldValue* As String, ByVal *newValue* As String) As String	Remplacer toutes les occurrences de la variable.
.Split(ByVal *separator* As Char) As String()	Permet de découper une chaîne selon un separator et retourne un tableau de chaîne.
.StartsWith(ByVal *value* As String) As Boolean	Vérifie si la variable commence par la chaîne value.
.Substring(ByVal *startIndex* As Integer, ByVal *length* As Integer) As String	Retourne la sous-chaîne de la variable.
.ToCharArray() As Char() .ToCharArray(ByVal *startIndex* As Integer, ByVal *length* As Integer) As Char()	Conversion de la variable string en tableau de caractères.
.ToLower() As String	Convertir en minuscule
.ToUpper() As String	Convertir en majuscule
.Trim() As String	Retirer tous les espaces du début et de la fin d'une chaîne de caractères.
.Trim(ByVal ParamArray *trimChars*() As Char) As String	Retirer tous les caractères du tableau *trimChars*() du début et de la fin d'une chaîne de caractères.

.Length() As Integer	Retourne le nombre de caractères de la variable.
.Chars(ByVal *index* As Integer) As Char	Retourne le caractère de la position index.
Date	
A terminer	

Conversion de type	
CSng	Conversion en type Single
CDbl	Conversion en type Double
CDec	Conversion en type Decimal
CByte	Conversion en type Byte
CShort	Conversion en type Short
CInt	Conversion en type Integer
CLng	Conversion en type Long
CStr	Conversion en type String
CChar	Conversion en Char

Fonctions mathématiques	
Type = Double. Decimal, Long, Byte, Short, Single	
System.Math.Abs(ByVal *value* As type) As Type	Valeur absolue
Type = double	
System.Math.Acos(ByVal *d* As type) As type System.Math.Asin(ByVal *d* As type) As type System.Math.Atan(ByVal *d* As type) As type	Arccosinus, Arcsinus Arctangente
Type = Decimal, Double	
System.Math.Ceiling(ByVal *d* As type) As type System.Math.Floor(ByVal *d* As type) As type System.Math.Round(ByVal *d* As type) As type	Arrondir à l'entier supérieur Arrondir à l'entier inférieur Arrondir à l'entier le plus près
Double	
System.Math.Cos(ByVal *d* As Double) As Double System.Math.Sin(ByVal *a* As Double) As Double System.Math.Tan(ByVal *a* As Double) As Double	Cosinus Sinus Tangente
System.Math.Exp(ByVal *d* As Double) As Double System.Math.Log(ByVal *d* As Double) As Double System.Math.Log(ByVal *a* As Double, ByVal *newBase* As Double) As Double System.Math.Log10(ByVal *d* As Double) As Double	Exposant (e^x) Logarithme en base e Logarithme en base newBase Logarithme en base 1
Type = Byte, Decimal, Double,Integer, Long, Short, Single	
System.Math.Max(ByVal *val1* As type, ByVal *val2* As type) As type Min(ByVal *val1* As type, ByVal *val2* As type) As type System.Math.Pow(ByVal *x* As Double, ByVal *y* As Double) As Double	Maximum de 2 nombres Minimum de 2 nombres Puissance (x^y)

Type = Decimal, double, Integer, Long, Short, Single	
System.Math. Sign(ByVal *value* As type) As Integer System.Math .Sqrt(ByVal *d* As Double) As Double	Signe (-1, 0, 1) Racine carrée

Opérateur Like Utilisation : *String* **Like** *Pattern* *String* : Chaîne à comparer *Pattern* : Patron à utiliser	
Symboles	
? ***** **#** [*liste*]	Un caractère Zéro ou plusieurs caractères Un chiffre (0–9) Liste de caractères
Liste de caractères	
[ABC] [!ABC] [A-C] [!A-C] [2468] [0-9]	Le caractère A, B ou C Un caractère qui n'est ni A, ni B, ni C Un caractère situé inclusivement entre A et C Un caractère qui n'est pas situé entre A et C Le chiffre 2, 4, 6 ou 8 (comme [!13579]) Un chiffre (caractère situé entre 0 ou 9) (comme #)

Format de l'affichage

FormatNumber (n,r)	Le nombre n est arrondi à r décimales
FormatCurrency(n,r)	Le nombre n est affiché avec un $ suivi du nombre selon FormatNumber (n,r)
FormatPercent (n,r)	Le nombre n est affiché en pourcentage et arrondi à r décimales

Tableau à une seule dimension	
nomTableau.Count	Nombre d'éléments
nomTableau.Max	Plus grande valeur
nomTableau.Min	Plus petite valeur
nomTableau.First	Premier élément
nomTableau.Last	Dernier élément
nomTableau.Average	Moyenne des éléments du tableau
nomTableau.Sum	Somme des éléments
numVar = Array.IndexOf(arrayName, value)	*numVar* l'indice de la première occurrence de *value* dans le tableau *arrayName*. -1 si la valeur n'est pas trouvée.
Tableau à plusieurs dimensions	
nomTableau.Rank()	retourne la dimension tableau.
nomTableau.GetUpperBound(x)	Retourne la taille de la dimension x

IO.File	
.ReadAllLines(ByVal *path* As String) As String()	Ouvre un fichier texte, lit toutes les lignes et ferme le fichier.
.WriteAllLines(ByVal *path* As String, ByVal *contents*() As String)	Créer un nouveau fichier, écrit le contenu du tableau contents() et ferme le fichier. Si le fichier existe, le contenu est modifié.
. OpenText(ByVal *path* As String) As System.IO.StreamReader	Ouvre un fichier et retourne un objet de type StreamReader.
.CreateText(ByVal *path* As String) As System.IO.StreamWriter	Créer ou ouvrir un fichier pour écriture. Retourne un objet de type StreamWriter.
.AppendText(ByVal *path* As String) As System.IO.StreamWriter	Créer un objet de type StreamWriter dont on peut ajouter des lignes à la fin du fichier.
.Exists(ByVal *path* As String) As Boolean	Vérifie si le fichier existe.
. Delete(ByVal *path* As String)	Supprime le fichier
.Copy(ByVal *sourceFileName* As String, ByVal *destFileName* As String)	Copie un fichier dans un nouveau fichier.
.Move(ByVal *sourceFileName* As String, ByVal *destFileName* As String)	Déplace un fichier.
.Open(ByVal *path* As String, ByVal *mode* As System.IO.FileMode) As System.IO.FileStream	Ouvre un fichier et retourne un fichier de type IO.FileStream.
.OpenRead(ByVal *path* As String) As System.IO.FileStream	Ouvre un fichier existant pour lecture et retourne un objet de type IO.FileStream. Ne vérifie pas si le fichier existe.
.OpenWrite(ByVal *path* As String) As System.IO.FileStream	Ouvre un fichier existant pour écriture et retourne un objet de type IO.FileStream. Ne vérifie pas si le fichier existe.

IO.FileMode est une variable qui peut prendre les valeurs suivantes	
Append	Ouvre un fichier et se positionne à la fin du fichier. Si le fichier n'existe pas, il est créé.
Create	Créer un nouveau fichier. Si le fichier existe, il est détruit pour en créer un nouveau.
CreateNew	Créer un nouveau fichier. Si le fichier existe déjà, une erreur est produite. (erreur = exception non vue dans le cours)
Open	Ouvre un fichier existant. Si le fichier n'existe pas, une erreur est produite. (erreur = exception non vue dans le cours)
OpenOrCreate	Ouvre un fichier existant ou en crée un nouveau si le fichier n'existe pas.
Truncate	Ouvre un fichier existant et efface son contenu. Si le fichier n'existe pas, une erreur est produite (exception).

IO.StreamReader	
. New(ByVal *path* As String)	Ouvre un fichier. Ne permet pas de vérifier l'existence du fichier.
. Close()	Fermeture du fichier
.Peek() As Integer	Retourne le prochain caractère, sans avancer la tête de lecture.
.Read() As Integer	Lit le prochain caractère
.ReadToEnd() As String	Lit de la position courante jusqu'à la fin du fichier
.ReadLine() As String	Lit la ligne courante
.EndOfStream() As Boolean	Indique si la position courante est la fin du fichier.
IO.StreamWriter	
.New(ByVal *path* As String)	Créer un nouveau fichier.
.Close()	Fermeture d'un fichier.
.WriteLine(ByVal *value* As Boolean)	Écrit la représentation texte d'un Boolean suivi d'un changement de ligne
.WriteLine(ByVal *buffer*() As Char)	Écrit le tableau de caractères, suivi par un changement de ligne.
.WriteLine(ByVal *value* As Char)	Écrit le caractère, suivi d'un changement de ligne.
. WriteLine(ByVal *value* As Decimal)	Écrit le décimal, suivi d'un changement de ligne.
. WriteLine(ByVal *value* As Double)	Écrit le double, suivi d'un changement de ligne.
. WriteLine(ByVal *value* As Integer)	Écrit l'integer, suivi d'un changement de ligne.
. WriteLine(ByVal *value* As Long)	Écrit le long, suivi d'un changement de ligne.
.WriteLine(ByVal *value* As Single)	Écrit le single, suivi d'un changement de ligne.
. WriteLine(ByVal *value* As String)	Écrit le string, suivi d'un changement de ligne.
. WriteLine()	Écrit juste un changement de ligne.
.Write(ByVal *value* As Boolean) .Write(ByVal *buffer*() As Char) .Write(ByVal *value* As Char) .Write(ByVal *value* As Decimal) .Write(ByVal *value* As Double) .Write(ByVal *value* As Integer) .Write(ByVal *value* As Long) .Write(ByVal *value* As Single) . Write(ByVal *value* As String)	Écrit les paramètres mais sans changement de ligne.

IO.FileStream	
.Seek(ByVal *offset* As Long, ByVal *origin* As System.IO.SeekOrigin) As Long	Positionner la tête de lecture et écriture d'un fichier. Retourne de nombre d'octets du début du fichier à la position courante.
.Name() As String	Retourne le nom du fichier.
.Position() As Long	Retourne de nombre d'octets du début du fichier à la position courante.

IO.SeekOrigin est une variable qui peut prendre les valeurs suivantes	
Begin	Se placer au début du fichier.
Current	Position courante du fichier.
End	Se placer à la fin du fichier.

IO.BinaryReader	
.Close()	Fermeture du fichier.
ReadBoolean() As Boolean ReadByte() As Byte ReadBytes(ByVal *count* As Integer) As Byte() ReadChar() As Char ReadChars(ByVal *count* As Integer) As Char() ReadDecimal() As Decimal ReadDouble() As Double ReadSingle() As Single ReadString() As String ReadInt16() As Short ReadInt32() As Integer ReadInt64() As Long	Lecture en binaire des types des valeurs de retour.
IO.BinaryWriter	
.Close()	Fermeture du fichier.
.Write(ByVal *value* As Boolean) Write(ByVal *buffer*() As Byte) Write(ByVal *buffer*() As Byte, ByVal *index* As Integer, ByVal *count* As Integer) Write(ByVal *value* As Byte) Write(ByVal *chars*() As Char) Write(ByVal *chars*() As Char, ByVal *index* As Integer, ByVal *count* As Integer) Write(ByVal *ch* As Char) Write(ByVal *chars*() As Char, ByVal *index* As Integer, ByVal *count* As Integer) Write(ByVal *ch* As Char) Write(ByVal *value* As Decimal) Write(ByVal *value* As Double) Write(ByVal *value* As Integer) Write(ByVal *value* As Long) Write(ByVal *value* As SByte) Write(ByVal *value* As Short) Write(ByVal *value* As Single) Write(ByVal *value* As String)	Écriture des paramètres de types différents