Création d'un algorithme

Analyse

input et output (I/O)

Squelette d'un programme

Création d'un algorithme

- Généralement, on ne crée pas un algorithme pour rien.
- On crée un algorithme pour résoudre un problème ou pour accomplir un objectif, automatiser, accélérer ou améliorer quelque chose, etc.
- La première étape consiste donc à ?

Analyser le problème

- C'est là que le pseudo-code nous sera utile.
- Le pseudo-code nous permet de définir les étapes de notre algorithme sans nous soucier du langage de programmation.
    - Les programmeurs d'expériences font cette étape dans leur tête, mais comme vous êtes encore en apprentissage, nous mettrons ces étapes sur papier.

Prenons un exemple simple.

- Je souhaite faire un programme qui récolte mon nom, la ville où j'habite et qu'une activité que j'aime et qui retournera ensuite un texte de présentation.
- Je dois donc effectuer les étapes suivantes:
    - Récolter l'information
    - Assembler l'information récolté dans une phrase de présentation
    - Afficher la phrase de présentation à l'intérieur d'un texte de présentation.
- Voyons ensemble à quoi ressemble le squelette d'un tel programme.

En pseudo-code

```
PROGRAMME DE PRÉSENTATION

ENTRÉES:
    string nom
    string ville
    string activitePrefere
SORTIE:
    string presentationComplete
DÉBUT
    ECRIRE "Entrez votre nom: "
    LIRE nom

    ECRIRE "Entrez le nom de la ville où vous habitez: "
    LIRE ville

    ECRIRE "Qu'elle est votre activité préféré? "
    LIRE activitePrefere

    presentationComplete <- "Je m'appelle " + nom + ", j'habite à " + ville + " et mon activité préféré est " + activitePrefere

    ECRIRE "Salut!"
    ECRIRE presentationComplete
    ECRIRE "Je suis ravit de vous rencontrer"
FIN
```

Remarquez bien les étapes

```
TITRE SIGNIFICATIF

ENTRÉES: <- 0 ou plusieurs entrées
    [type de variable (plus d'information au prochain cours)] nomDeLaVariable <- syntaxe lowerCamelCase, nom significatif

SORTIE: <- 1 ou plusieurs sorties
    [type de variable] nomDeLaVariable

DEBUT
    Écrire ici toutes les étapes pour passer des entrées aux sorties
FIN
```

À votre tour.

- Ouvrez un éditeur de texte de votre choix.
- Écrivez l'algorithme en pseudo-code d'un programme qui fusionne 4 chaines de caractère avec un séparateur.
    - Par exemple: Je veux pouvoir écrire: "salut" "est-ce" "que" "ça vas", et choisir le séparateur "|". L'algorithme doit ensuite me retourner le résultat: "salut|est-ce|que|ça vas"

Shuuuttt!! Exercice en cours...

Traduction d'un algorithme en C#

Output

Variable

Input

Output

- Pour afficher de l'information dans la console, l'équivalent de la commande ÉCRIRE en pseudo-code. Il y deux possibilitées:
  - `Console.WriteLine("Allo");` <- cette syntaxe écrit le texte entre les guillemets et effectue ensuite un saut de ligne.
  - `Console.Write("Allo");` <- Cette syntaxe écrit les textes entre les guillemets sans effectuer de saut de ligne.
- Ainsi ces instructions:

```
Console.WriteLine("Allo, classe de prog1!");
Console.Write("Est-ce que, ");
Console.Write("ça va bien?");
```

- Afficheront

```
Allo, classe de prog1!
Est-ce que, ça va bien?
```

Pour la suite, nous aurons besoin de variable... Mais qu'est-ce donc que cela?

Variable

- La mémoire vive ou RAM (Random Access Memory) contient les programmes informatiques et les données traitées pendant leurs exécutions
- Pour stocker une valeur en mémoire dans la RAM, nous devons y réserver un espace. Cela ce fait avec la déclaration de variable.
- En pseudo-code, lorsqu'on défini les entrées et sorties, nous définissons en faite les espaces mémoires à réserver dans la RAM qui seront utilisé pendant l'exécution de notre programme.
- Le type de donnée permet de définir la quantité de mémoire à réserver. (plus de détail sur ça au prochain cours)

Syntaxe en C#

- Si on reprend notre exemple précédent:

```
ENTRÉES:
    string nom
    string ville
    string activitePrefere
SORTIE:
    string presentationComplete
```

- Cela donnerait:

```
// ENTRÉES:  les // servent à définir la ligne comme un commentaire.
    string nom;  // Ne pas oublier le ; à la fin de chaque instruction en C#
    string ville;
    string activitePrefere;
// SORTIE:
    string presentationComplete;
```

Opérateur d'affectation de variable

- En c# (et dans la majorité des langages de programmation), l'instruction `=` sert à affecter une valeur dans une variable.
- Par exemple:

```
string nom;
nom = "Kevin Bessette";
Console.WriteLine(nom);
```

- Affichera dans la console `Kevin Bessette`
- *Notez l'utilisation des guillemets "Kevin Bessette" lorsqu'on veut définir une chaîne de caractère (string).*

Opérateur de concaténation de chaine

- En c# (et dans la majorité des langages de programmation), l'instruction `+`, lorsque utilisé sur des chaînes de caractère, effectue une concaténation.
- Par exemple:

```
string nom;
string prenom;
nom = "Bessette";
prenom = "Kevin"
Console.WriteLine(nom + ", " + prenom);
```

- Affichera dans la console `Bessette, Kevin`

Input

- Plutôt que d'affecter manuellement une valeur à une variable. Il est fréquent que le programme permette à l'utilisateur de choisir la valeur à affecter.
    - L'équivalent de l'instruction LIRE en pseudo-code.
- En c# on utilise l'instruction `Console.ReadLine();`
- Par exemple, instruction:

```
string nom;
nom = Console.ReadLine();
```

- permet à l'utilisateur d'écrire une chaîne de caractère qui sera ensuite affecté à la variable nom.

Alogithme final

```
// PROGRAMME DE PRÉSENTATION
// ENTRÉES:
    string nom;
    string ville;
    string activitePrefere;
// SORTIE:
    string presentationComplete;
// DÉBUT
    Console.WriteLine("Entrez votre nom: ");
    nom = Console.ReadLine();

    Console.WriteLine("Entrez le nom de la ville où vous habitez: ");
    ville = Console.ReadLine();

    Console.WriteLine("Qu'elle est votre activité préféré? ");
    activitePrefere = Console.ReadLine();

    presentationComplete = "Je m'appelle " + nom + ", j'habite à " + ville + " et mon activité préféré est " + activitePrefere;

    Console.WriteLine("Salut!");
    Console.WriteLine(presentation complete);
    Console.WriteLine("Je suis ravit de vous rencontrer");
// FIN
```

Quiz

Quelle est la différence entre `Console.WriteLine("Bonjour");` et `Console.Write("Bonjour)?;`

`Console.WriteLine("Bonjour");` ajoutera un saut de ligne à la fin.

Quelle est la commande pour permettre à l'utilisateur de l'application d'inscrire une valeur?

`Console.ReadLine();`

Mais assurez-vous de stocker cette valeur quelque part pour pouvoir la réutiliser

```
string name;
name = Console.ReadLine();
```