TP03 - Rendre la monnaie de sa pièce

1 Introduction

Créer un programme est un acte qui nécessite :

- De la logique : c'est la capacité à envisager tous les cas de figures. Cette logique "élémentaire Watson" permet de comprendre et décrire ce que d'autres voient comme étant instinctif.
- Des notions artistiques : un code doit être "beau", à la manière de Saint-Exupéry, "*La perfection est atteinte, non pas lorsqu'il n'y a plus rien à ajouter, mais lorsqu'il n'y a plus rien à retirer*". Nous verrons plus tard l'acronyme KISS que les anglo-saxons résument par "Keep It Simple and Stupid".
- De la mémoire : en effet, pendant le processus de codage, le programmeur génère de nouvelles variables, imagine des fonctions qui ne sont pas encore écrites, mais il utilise déjà leur fonctionnement. Il faut alors savoir garder en tête des modifications à faire plus tard, pour terminer celles en cours…

Cette séance est donc un apprentissage de la pensée, une manière de révéler l'envers du décor.

Vous incarnez Watson, au XXIe siècle : vous n'êtes pas médecin mais docteur… en informatique !

2 Objectifs

Dans ce TD, l'objectif principal est :

- se familiariser avec les instructions conditionnelles

les objectifs secondaires sont :

- la capacité à corriger son code,
- la capacité à décrire sa pensée,
- la faculté d'envisager l'ensemble des solutions

3 ÉNONCÉ DU PROBLÈME : RENDRE LA MONNAIE

Depuis des années, nous voyons les commerçants inscrire le montant de l'achat, la monnaie donnée par le client et l'afficheur indique la somme à rendre.

Jusque-là, une simple soustraction suffit, mais cela devient intéressant lorsque le commerçant rend la monnaie : comment calcule-t-il ? Quel est son mode de pensée ?

3.1 INDICE N°1

La monnaie de tous les pays est généralement constituée de nombres précis : il n'existe pas de pièces de 23 centimes. Commencez par chercher de ce côté-là Watson…

3.2 INDICE N°2

Il y a une notion de modulo : en C#, le symbole % est utilisé.

Par exemple, si une grande boite de donuts contient 10 donuts, et une petite boite en contient 3, combien de boite faudra-t-il pour contenir 126 donuts (sans vide dans les grandes boites) ?

Pour les grandes boites : 126 ÷ 10 = 12

Il y a alors deux possibilités :

1. Calculer 126 – (10*12)
2. Calculer 126 % 10

3.3 STRUCTURE DU PROGRAMME

Le programme devra demander

- le montant total des achats
- la monnaie donnée par le client (par exemple 14 signifiera 14 euros. Pas de centimes)

Et affichera le nombre de pièces de chaque valeur, et le nombre de billets de chaque valeur.

3.4 JEUX DE TEST

Pour valider votre programme, voici quelques exemples de valeurs

Montant Total	Monnaie donnée
100 €	40 €
50 €	50 €
50 €	60 €
50 €	80 €

Ne lisez pas la page suivante avant d'avoir réfléchi au problème

4 La réponse de Sherlock Holmes

La solution est on ne peut plus simple pour moi, Watson… continuez à réfléchir !

4.1 Un peu d'aide quand même ?

Quoique, pour cette première enquête, je vais vous aider :

La monnaie fiduciaire existe depuis de nombreuses années et doit permettre, par addition, de couvrir toutes les valeurs avec le minimum d'éléments (pièces ou billets).

A part lorsque, fâché, vous avez payé vos impôts en pièces de 10 centimes, la plupart des gens normalement constitué souhaite minimiser l'encombrement dans leur porte-monnaie.

Donc, le système retenu est basé sur les nombres 1, 2, 5 et leurs multiples : il n'existe pas de billet de 300 € ou de pièce de 4 €.

Il faut donc rendre la monnaie en commençant par la valeur la plus élevée mais qui soit inférieure à la différence entre le montant total et la monnaie donnée par le client :

1. MonnaieRendue = MonnaieClient – MontantTotal

Il faut déjà s'assurer que le client propose une somme supérieure ou égale au montant total :

- Si MonnaieRendue est inférieure à zéro, le commerçant doit refuser d'aller plus loin avec le client et doit proposer de laisser des articles ou de donner plus de monnaies. Le plus simple est donc de tout recommencer.
- Si MonnaieRendue est égale à zéro, c'est que le client à fait l'appoint… la machine doit accepter la transaction
- C'est seulement si la monnaie rendue est supérieure à zéro que le commerçant devra rendre de l'argent.

4.2 QUE RENDRE ?

Watson, vous le faites exprès ?!?

Il faut tester la plus grande valeur, le billet de 500 €, sinon le billet de 200 €, puis le billet de 100 €.

Vous pourriez faire une liste ? Voici une notion nouvelle, le tableau

```
let Monnaie = { 500, 200, 100, 50, 20, 10, 5, 2, 1 };
```

Pour lire la liste, il faut une boucle ou répétition : ici, la boucle for fera l'affaire, car nous allons proposer de rendre un nombre de billets et pièces dans la liste.

Il faut diviser MonnaieRendue par la valeur la plus élevée et conserver ce nombre.

Bien entendu, il faut alors afficher la valeur puis la soustraire à MonnaieRendue

- Console.Log ("Rendez "+Nombre+" fois "+Monnaie[index]+" €") ;
- MonnaieRendue = MonnaieRendue – Nombre * Monnaie[index] ;

Mais il faut recommencer pour toutes les valeurs possibles… sauf si MonnaieRendue est déjà à zéro (instruction break ;).