TD N° 2 : POO

Exercice 1 : Détection de clients à risque

Une entreprise souhaite analyser ses clients afin de détecter les profils a risque.

Chaque **commande** est caractérisée par: un montant, un statut (PAID, PENDING), un nombre de jours de retard.
Chaque **client** est caractérisé par : un nom, un secteur, un tableau de commandes [].

Un client est considéré à risque si au moins une des conditions suivantes est vérifiée :

- le montant total des commandes en retard dépasse 20000
- le nombre de commandes en attente (PENDING) est supérieur ou égal a 3
- le montant paye est inferieur a 50% du montant total commande

Travail demandé :

1. Implémenter une fonction constructeur order
2. Implémenter une fonction constructeur customer
3. Implémenter les méthodes suivantes :
   - getTotalCommande()
   - getTotalPaye()
   - countEnAttente()
   - getMontantRetard()
   - isRisque()
4. Créer un jeu de données avec plusieurs clients
5. Afficher les clients a risque

========== CORRECTION ==========

```
<script type="text/javascript">

function getTotalCommande() {
    var s = 0;
    for (var i = 0; i < this.commandes.length; i++) {
         s = s  + this.commandes[i].montant;
    }
    return s;
}

function getTotalPaye() {
    var s = 0;
    for (var i = 0; i < this.commandes.length; i++) {
```

```
    if(this.commandes[i].statut ==='PAID')
         s = s  + this.commandes[i].montant;
    }
    return s;
}

function countEnAttente(){
    var c = 0;
    for (var i = 0; i < this.commandes.length; i++) {
         if(this.commandes[i].statut ==='PENDING')
             c = c + 1;
    }
    return c;
}

function getMontantRetard() {
    var s = 0;
    for (var i = 0; i < this.commandes.length; i++) {
         if(this.commandes[i].joursRetard > 0)
             s = s  + this.commandes[i].montant;
    }
    return s;
}

function isRisque(){
    var montantRetard = this.getMontantRetard();
    var nombreRetard = this.countEnAttente();
    var totalCommande = this.getTotalCommande();
    var totalPaye = this.getTotalPaye();
    if(montantRetard>20000 || nombreRetard>= 3 || totalPaye  < 0.5*totalCommande)
           return true;

    return false;
}

function afficherCustomer(){
    document.write('Nom : '+ this.nom + '<br>');
    document.write('Secteur : '+ this.secteur + '<br>');
    document.write('Total Commandes : '+ this.getTotalCommande() + '<br>');
    document.write('Risque : '+ this.isRisque() + '<br>');
}


function Order(montant, statut, joursRetard){

    // les attributs
    this.montant = montant;
    this.statut = statut;
```

```
    this.joursRetard = joursRetard;

}

function Customer(nom, secteur, commandes){

    // les attributs
    this.nom = nom;
    this.secteur = secteur;
    this.commandes = commandes;

    // les méthodes
    this.getTotalCommande = getTotalCommande;
    this.getTotalPaye = getTotalPaye;
    this.countEnAttente = countEnAttente;
    this.getMontantRetard = getMontantRetard;
    this.isRisque = isRisque;
    this.afficherCustomer = afficherCustomer;
}


// Jeu de données
var commandes1 = [
    new Order(1000, 'PAID', 0),
    new Order(2000, 'PENDING', 5)
];
var c1 = new Customer('Ahmed', 'IT', commandes1)

var commandes2 = [
    new Order(800, 'PAID', 2)
];
var c2 = new Customer('Asmae', 'Commerce', commandes2);

var clients = [c1, c2];

for (var i = 0; i < clients.length; i++) {
    document.write('<br>Client num ' + (i+1) + ' =====   <br>');
    clients[i].afficherCustomer()
}

</script>
```

Exercice 2 : Contrôle de factures fournisseurs

Une entreprise souhaite automatiser le contrôle de ses factures fournisseurs.

Chaque **ligne de facture** contient : une quantité, un prix unitaire, une catégorie (PRODUCT, SERVICE)
Chaque **facture** contient : un numéro, un tableau de lignes [], un taux de TVA et un statut

Règles de gestion :

- une facture est invalide si une ligne contient une quantité <= 0 ou un prix < 0
- une facture est suspecte si :
  - son total HT dépasse 100000
  - ou elle contient plus de 5 lignes de type SERVICE
- une facture est bloquée si son statut est DRAFT ou CANCELLED

Travail demandé :

1. Implémenter une fonction constructeur invoiceLine
2. Implémenter une fonction constructeur invoice
3. Implémenter les méthodes suivantes :
   - getTotalHT()
   - compterParCategorie(categorie)
   - isInvalide()
   - isSuspecte()
   - isBloquee()
4. Afficher les factures suspectes

========== CORRECTION ==========

```
<script type="text/javascript">

function getTotalLigne() {
    return this.quantite * this.prixUnitaire;
}

function getTotalHT() {
    var i = 0;
    var total = 0;
    for (i = 0; i < this.lignes.length; i++) {
        total = total + this.lignes[i].getTotalLigne();
    }
    return total;
}

function countByCategorie(categorie) {
    var i = 0;
    var nombre = 0;
    for (i = 0; i < this.lignes.length; i++) {
        if (this.lignes[i].categorie == categorie) {
```

```
            nombre = nombre + 1;
        }
    }
    return nombre;
}

function isInvalide() {
    var i = 0;
    for (i = 0; i < this.lignes.length; i++) {
        if (this.lignes[i].quantite <= 0 || this.lignes[i].prixUnitaire < 0) {
            return true;
        }
    }
    return false;
}

function isSuspecte() {
    if (this.getTotalHT() > 100000) {
        return true;
    }
    if (this.countByCategorie("SERVICE") > 5) {
        return true;
    }
    return false;
}

function isBloquee() {
    if (this.statut == "DRAFT" || this.statut == "CANCELLED") {
        return true;
    }
    return false;
}

function afficherFacture() {
    document.write("<h2>Exercice 2 - Facture</h2>");
    document.write("Numéro : " + this.numero + "<br>");
    document.write("Total HT : " + this.getTotalHT() + "<br>");
    document.write("Nombre de lignes SERVICE : " +
this.countByCategorie("SERVICE") + "<br>");
    document.write("Facture invalide : " + this.isInvalide() + "<br>");
    document.write("Facture suspecte : " + this.isSuspecte() + "<br>");
    document.write("Facture bloquée : " + this.isBloquee() + "<br><br>");
}

function ligneFacture(quantite, prixUnitaire, categorie) {
    // les attributs
    this.quantite = quantite;
    this.prixUnitaire = prixUnitaire;
```

```
    this.categorie = categorie;
    this.getTotalLigne = getTotalLigne;
}

function facture(numero, lignes, tauxTVA, statut) {

    // les attributs
    this.numero = numero;
    this.lignes = lignes;
    this.tauxTVA = tauxTVA;
    this.statut = statut;

    // les méthodes
    this.getTotalHT = getTotalHT;
    this.countByCategorie = countByCategorie;
    this.isInvalide = isInvalide;
    this.isSuspecte = isSuspecte;
    this.isBloquee = isBloquee;
    this.afficherFacture = afficherFacture;
}

</script>
```