Projet de POO à remettre
Programmation en java

NB : le travail doit être fait par binôme (ou monôme).

Le travail doit être imprimé et remis (rapport + CD), le mardi 15/05/2018

Le travail doit être envoyé à l'adresse recupspace@gmail.com en précisant dans l'objet : ProjetPOO + noms des étudiants

Les programmes écrits doivent être lisibles et bien commentés.

Les copiages seront sévèrement sanctionnés

Dans ce TP, on désire gérer les clients d'un d'opérateur téléphonique. Un opérateur est décrit par un nom et un ensemble de points de vente, un pourcentage de couverture par wilaya (par exemple 100% à Alger et 40% à Tebessa) et possède un ensemble de clients (abonnés).

Un point de vente est décrit par un nom d'agence (String), un type (principal ou secondaire), une adresse (objet) et un numéro de téléphone (composé de 10 chiffres et commençant obligatoirement par un zéro).

Un client est décrit par un numéro de téléphone (composé de 10 chiffres et commençant obligatoirement par un zéro), un type d'abonnement (forfaitaire, prépayé ou libre), un numéro de contrat (String), une date de contrat, un nom, un prénom, une adresse (objet) et une adresse mail.

Pour simplifier l'application, on s'intéresse uniquement aux appels et aux SMS.

Les trois types d'abonnement diffèrent selon le mode de paiement.

Un abonnement libre est payé tous les deux mois sur la base d'une facture établie ; le montant de la facture est égal au forfait fixe (500 DA) plus le montant des appels effectués et des SMS envoyés plus un pourcentage de TVA (19% du montant). Le montant d'un appel est égal au nombre d'unités * tarif de l'unité (une unité = une minute). Pour simplifier, on suppose que le tarif de l'unité est égal à 4DA pour un appel au sein de l'opérateur, 5DA pour un appel vers autre opérateur local et 20DA pour un appel vers l'étranger.

Un abonnement forfaitaire possède une durée de forfait (par exemple 2 mois) et un montant de forfait (par exemple 2300DA) à consommer dans les appels et SMS. Quand le montant du forfait est épuisé, le client ne peut que recevoir des appels ou des SMS. S'il essaye d'appeler ou d'envoyer, une exception « Solde insuffisant » devra être générée.

Le type prépayé utilise des cartes de recharge. Une carte de recharge est décrite par un numéro de série (String sur 14 chiffres), une date de validité, un montant et un état (activé ou non). Lorsqu'un client recharge son compte, on enregistre le numéro de la carte payée et la date de son activation et on change son état. Le client peut charger plusieurs cartes et le montant du solde est alors cumulé (additionné). On peut consulter

le solde restant et la date limite de validité de la recharge (par exemple 3 mois). Quand le montant de la recharge est épuisé, le client ne peut que recevoir.

Un client peut effectuer un ensemble d'appels et recevoir un ensemble d'appels (nombre illimité). Un appel émis (sortant) possède un numéro de destinataire, une date, une heure, une durée et un montant (calculé) qui sera déduit du solde restant (dans le cas d'un abonnement prépayé ou forfaitaire) ou ajouté au cumul pour la facturation (dans le cas d'un abonnement libre). Un appel reçu (entrant) possède un numéro d'appelant, une date, une heure et une durée. Les informations concernant les appels et les SMS sont enregistrées pour chaque numéro de client. Pour un numéro donné, on peut calculer la durée cumulée des appels entrants et la durée cumulée des appels sortants et les afficher à tout moment.

Un client peut envoyer un SMS qui est décrit par le numéro de l'expéditeur, le numéro du destinataire, date et heure d'envoi, statut (envoyé, reçu, echec,…) et le texte du message.

Chaque numéro de client peut avoir un bonus : les bonus sont de différentes catégories : nombre d'heures, nombre de SMS, solde additionnel. Un bonus possède une date limite de consommation ; au delà de cette date, il est repris ; il diminue à chaque consommation.

Un numéro de client peut être bloqué suite au non rechargement ou défaut de paiement de la facture pendant une durée limitée et fixée au préalable (par exemple 90 jours pour les recharges et 30 jours pour les factures). Le numéro est relancé trois fois par SMS avant de décider de le bloquer.

Questions

Donner une description des classes nécessaires à cette application en maximisant la réutilisation et la modularité. Utilisez aussi l'héritage (voir les indications ci-dessous).

L'application doit pouvoir répondre aux requêtes suivantes :

Remplissage des données (utiliser les collections)

Ecrire une méthode Remplir pour la saisie automatique des données

Pour l'opérateur : On devra introduire (par programme) des valeurs au choix (prévoir 7 points de vente et 5 wilayas au minimum)

Pour les clients : On devra introduire (par programme) des valeurs au choix comme paramètres des différents constructeurs (20 clients de différents types, des appels entrants/sortants, des SMS, …)

Mise à jour des données

Pour l'opérateur : ajouter/ supprimer /modifier un point de vente (changer type ou numéro ou adresse), changer pourcentage de couverture pour une wilaya donnée.

Pour les clients : ajouter /modifier / bloquer un client ; pour l'ajout le type d'abonnement devra être précisé.

Ajouter un appel entrant ou sortant, ajouter un SMS entrant ou sortant… pour un numéro donné

Consultation des données

Pour l'opérateur :

Afficher les informations concernant l'opérateur

Pour les clients

- La liste des clients par type d'abonnement
- La liste des numéros bloqués, on affichera en plus la date de blocage et le motif du blocage.
- La liste des numéros relancés pour paiement ou rechargement avec les dates de rappel
- La liste des clients par wilaya (selon l'adresse du client, l'adresse devra comporter un attribut wilaya)
- Recherche d'un numéro donné et afficher ses informations
- Recherche d'un numéro donné et afficher les appels entrants et sortants et les durées cumulées
- Etablir facture pour un numéro donné
- Afficher tous les numéros arrivés à échéance de paiement
- Afficher toutes les factures (associées aux numéros) en instance de paiement

Organisation de l'application

L'application devra s'exécuter via un menu affiché à l'utilisateur comportant les choix suivants :

1. Remplissage automatique des données
2. Gestion de l'opérateur
3. Gestion des clients (numéros)
4. Gestion des factures
5. Gestion des bonus
6. Quitter

A chaque choix correspond un sous-menu (par exemple pour Gestion des clients)

1. Ajouter numéro
2. Modifier numéro (modifier adresse, débloquer)
3. Supprimer numéro
4. Afficher la collection des clients par type d'abonnement

… (voir les options citées plus haut)

. Quitter

Gestion des factures

1. Etablir facture pour un numéro donné
2. Afficher tous les numéros arrivés à échéance de paiement
3. Toutes les factures en instance de paiement
4. Relancer les numéros pour les rechargements/ paiements
5. Quitter

Gestion des bonus

1. Affecter bonus à des clients (on précisera le type de bonus) 2. Afficher les clients ayant bénéficié de bonus 3. Quitter

Indications

- Prévoir dans chaque classe, un constructeur avec paramètres qui permettra de préparer le remplissage des données par programme (pour les besoins de test) et un constructeur sans paramètres.
- Respecter le principe d'encapsulation dans les classes, les attributs doivent être déclarés « private » et prévoir les accesseurs nécessaires.
- Utiliser les **types énumérés** pour toutes les classes d'énumération
- Pour chaque classe, prévoir des ***constructeurs*** et une méthode **Afficher** (redéfinir la méthode **toString)**, en plus des méthodes nécessaires à la manipulation des objets.
- Prévoir l'utilisation d'**exceptions** pour les traitements d'erreurs.
- Les affichages doivent être lisibles (bien alignés/ sauts de lignes nécessaires, noms des champs) cohérents et explicites.
- Les noms des classes, des attributs et des méthodes doivent être significatifs et non ambigus.
- Le code java doit être bien commenté.

Rapport à remettre

Le rapport devra comporter :

- Une description des classes avec les liens entre elles.
- Pour chaque classe, on précisera les attributs avec leurs types et les signatures des méthodes avec une brève description de chaque méthode.
- Une description sommaire de la classe programme