4 amis partent en vacances.

Résumé

Isabelle, Philippe, Maxime et Eric partent en vacances de neige. Pour se simplifier la vie, ils ont décidé de partager les frais. C'est Philippe qui fera les comptes (il a un DUT Info et il est donc le plus fort en math de tous. Qui plus est, il frime toujours avec son tout nouveau portable android. C'est l'occasion idéale de rentabiliser son investissement. Vous allez l'aider à utiliser son bijou high tech ...

1 Rendu.

Le résultat de votre travail devra être remis, étape par étape, via moodle. Les étapes seront mentionnée dans le sujet sous cette forme :

Etape 0: listes des fichiers à remettre..

2 Problème.

Quand un achat commun sera fait, Philippe enregistrera le montant de l'achat et qui l'a financé. A la fin des vacances les amis pourront ainsi régulariser leur situation financière et se quitter (ou pas) on bons amis.

Exemple : Isabelle a payé le transport pour 1000 €et Maxime l'hébergement-restauration pour 800 €. Au final nous aurons donc les comptes suivants :
Dépenses totales : 1800 €, soit 450 €par personne.
Isabelle doit recevoir : 1000 - 450 = 550 €
Maxime doit recevoir : 800 - 450 = 350 €
Philippe et Eric doivent rembourser 450 €chacun.

Pratiquement, pour Philippe, il suffit de tenir à jour le compte de ce que chacun doit rembourser ou recevoir. Chaque fois que quelq'un paye pour le groupe, son compte est crédité des 3/4 du montant tandis que chacun des comptes de ses trois amis est débité du 1/4 du même montant. Nous aurions alors le même calcul que précédemment, sous une autre forme :

	Isabelle	Philippe	Maxime	Eric
Initial	0	0	0	0
transport	0 + 750 = 750	0 -250 = -250	0 -250 = -250	0 -250 = -250
Hébergement	750 - 200 = 550	-250 - 200 = -450	-250 + 600 = 350	-250 -200 = -450
Final	550	-450	350	-450

IL reste à réaliser le programme qui réalise tout ceci de manière ergonomique.

3 Niveau 1 : Réalisation basique.

Pour vous aider, trois fichiers vous sont fournis :

1. Un fichier de ressources (`strings.xml`).
2. Le layout de départ (`activity_main.xml`).
   Celui ci affiche un tableau ayant pour colonnes le nom du participant, une zone de saisie et une zone d'affichage du solde, à raison d'une ligne par participant. Sous ce tableau figure un bouton permettant de prendre en compte les saisies.
3. Le modèle à utiliser (`Modele.java`). Il dispose des méthodes suivantes :
   - Le constructeur (avec un tableau de chaînes en paramètre : les noms des participants.
   - La méthode `update` (avec deux paramètres : le numéro du participant (de 0 à 3) et le montant payé (1000 pour Isabelle par exemple). Alternativement, on peut remplacer le numéro du participant par son nom.
   - La méthode `getSolde` (avec comme paramètre unique la numéro du participant ou son nom) qui permet d'obtenir le solde actuel de ce participant.

Q 1. Créez un projet et intégrer y les fichiers cités précédemment.
Vous devriez alors obtenir un projet sans erreur de compilation mais non fonctionnel.
Q 2. Remplacez Eric par Jean.
Q 3. Ajouter la création du modèle lors de la phase d'initialisation de l'activité. Vous pouvez au choix obtenir les noms des participants (à fournir au constructeur) soit depuis les ressources, soit depuis les composants visuels.
Le traitement de la mise à jour s'effectue de la façon suivante :

1. Les quatre champs de saisie sont convertis en réel. Lors de l'opération, on compte le nombre de champs non nuls. les saisies non conformes sont considérées comme nulles.
2. Si le nombre de champs non nuls est différent de 1, il y a une erreur de saisie. Le traitement de l'action s ?arrête.

3. Si il y a une saisie non nulle et une seule, on demande au modèle de mettre à jour les comptes des participants.
4. Il faudra pour terminer mettre à jour les soldes affichés des participants en fonction de la valeur nouvellement calculée par le modèle.

Q 4. Ecrire le traitement des mise à jour.

Etape 1: Remettez la totalité du projet (en un seul fichier compressé)..

4 Transaction entre participants.

Ponctuellement, il est possible qu'un participant prête une somme d'argent à un autre participant. Il est facilement de modifier le programme pour pouvoir enregistrer cette transaction et la répercuter sur le décompte final. (Maxime prête 10 €à Isabelle) Il suffit en effet de créditer le prêteur (Maxime) et de débiter le bénéficiaire (Isabelle) de la somme correspondante. Sur l'écran, on peut ainsi saisir +10 sur le ligne Maxime et -10 sur la ligne Isabelle. Informatiquement, ceci se caractérise par le fait que plusieurs valeurs saisies sont non nulles.

Q 5. Modifiez le programme pour que celui-ci traite le scénario des transactions entre participants.

Etape 2: MainActivity.java.

5 Niveau 2 : Utilisation d'un Spinner

On veut tester une autre IHM où pour le premier scénario, le choix du participant payeur (unique) est effectué via un spinner. Vous avez la latitude de modifier, si vous le souhaitez, la façon dont sont enregistrés les noms des participants dans les ressources. Il existe toutefois d'autres solutions.

Ajouter les composants nécessaires (Spinner, zone de saisie) à l'interface existante.
Modifier le code de l'activité pour gérer cette nouvelle fonctionnalité. Vous pouvez (toujours si vous le souhaitez) créer de nouvelles classes.

Etape 3: Remettez la totalité du projet (en un seul fichier compressé)..

6 Niveau 3 : Utilisation d'une base de données.

Complétez le projet pour que chaque opération soit enregistrée dans une table. A vous de construire le schéma relationnel ad hoc.

Remarque : Pour être cohérent,, il faudrait aussi, à ce niveau du programme, rangez les noms des participants dans une autre table. Le TP étant en temps limité, nous n'auront hélas pas l'opportunité de le faire.

Etape 4: Remettez la totalité du projet (en un seul fichier compressé)..