1. Develop a simple banking system that allows users to create accounts, deposit money, withdraw money, and check balance. Implement methods for account creation, deposit, withdrawal, and balance inquiry.

**Methods**:

* createAccount(String accountHolderName, double initialDeposit)
* depositMoney(String accountNumber, double amount)
* withdrawMoney(String accountNumber, double amount)
* checkBalance(String accountNumber)

program:

import java.util.HashMap;

import java.util.Map;

import java.util.UUID;

public class BankingSystem {

private static class Account {

String accountNumber;

String accountHolderName;

double balance;

public Account(String accountHolderName, double initialDeposit) {

this.accountNumber = UUID.randomUUID().toString();

this.accountHolderName = accountHolderName;

this.balance = initialDeposit;

}

public void deposit(double amount) {

balance += amount;

}

public boolean withdraw(double amount) {

if (amount <= balance) {

balance -= amount;

return true;

} else {

return false;

}

}

public double getBalance() {

return balance;

}

@Override

public String toString() {

return "Account Number: " + accountNumber + ", Account Holder: " + accountHolderName + ", Balance: " + balance;

}

}

private Map<String, Account> accounts;

public BankingSystem() {

this.accounts = new HashMap<>();

}

public String createAccount(String accountHolderName, double initialDeposit) {

Account newAccount = new Account(accountHolderName, initialDeposit);

accounts.put(newAccount.accountNumber, newAccount);

return newAccount.accountNumber;

}

public void depositMoney(String accountNumber, double amount) {

Account account = accounts.get(accountNumber);

if (account != null) {

account.deposit(amount);

System.out.println("Deposited " + amount + " to account " + accountNumber);

} else {

System.out.println("Account not found!");

}

}

public void withdrawMoney(String accountNumber, double amount) {

Account account = accounts.get(accountNumber);

if (account != null) {

if (account.withdraw(amount)) {

System.out.println("Withdrew " + amount + " from account " + accountNumber);

} else {

System.out.println("Insufficient balance!");

}

} else {

System.out.println("Account not found!");

}

}

public void checkBalance(String accountNumber) {

Account account = accounts.get(accountNumber);

if (account != null) {

System.out.println("Balance for account " + accountNumber + ": " + account.getBalance());

} else {

System.out.println("Account not found!");

}

}

public static void main(String[] args) {

BankingSystem bank = new BankingSystem();

String account1 = bank.createAccount("John Doe", 1000.0);

String account2 = bank.createAccount("Jane Smith", 500.0);

bank.depositMoney(account1, 200.0);

bank.withdrawMoney(account1, 150.0);

bank.checkBalance(account1);

bank.depositMoney(account2, 300.0);

bank.withdrawMoney(account2, 700.0);

bank.checkBalance(account2);

}

}

**Output:**

**java -cp /tmp/0EfNN32NNC/BankingSystem**

**Deposited 200.0 to account 07023950-b723-4ca1-9137-6e780b07e909**

**Withdrew 150.0 from account 07023950-b723-4ca1-9137-6e780b07e909**

**Balance for account 07023950-b723-4ca1-9137-6e780b07e909: 1050.0**

**Deposited 300.0 to account 5f6c5f69-a648-4efd-9129-a146633a8c6a**

**Withdrew 700.0 from account 5f6c5f69-a648-4efd-9129-a146633a8c6a**

**Balance for account 5f6c5f69-a648-4efd-9129-a146633a8c6a: 100.0**

**=== Code Execution Successful ===**

2. Create an expense tracker that allows users to add expenses, categorize them, and view a summary report. Implement methods to add expenses, categorize expenses, and generate reports.

**Methods**:

* addExpense(String description, double amount, String category)
* viewExpensesByCategory(String category)
* generateExpenseReport()

**program:**

import java.util.ArrayList;

import java.util.HashMap;

import java.util.List;

import java.util.Map;

public class ExpenseTracker {

private static class Expense {

String description;

double amount;

String category;

public Expense(String description, double amount, String category) {

this.description = description;

this.amount = amount;

this.category = category;

}

@Override

public String toString() {

return "Description: " + description + ", Amount: " + amount + ", Category: " + category;

}

}

private List<Expense> expenses;

public ExpenseTracker() {

this.expenses = new ArrayList<>();

}

public void addExpense(String description, double amount, String category) {

expenses.add(new Expense(description, amount, category));

}

public void viewExpensesByCategory(String category) {

System.out.println("Expenses for category: " + category);

for (Expense expense : expenses) {

if (expense.category.equalsIgnoreCase(category)) {

System.out.println(expense);

}

}

}

public void generateExpenseReport() {

Map<String, Double> categoryTotals = new HashMap<>();

double totalExpenses = 0;

for (Expense expense : expenses) {

categoryTotals.put(expense.category, categoryTotals.getOrDefault(expense.category, 0.0) + expense.amount);

totalExpenses += expense.amount;

}

System.out.println("Expense Report:");

for (Map.Entry<String, Double> entry : categoryTotals.entrySet()) {

System.out.println("Category: " + entry.getKey() + ", Total: " + entry.getValue());

}

System.out.println("Total Expenses: " + totalExpenses);

}

public static void main(String[] args) {

ExpenseTracker tracker = new ExpenseTracker();

tracker.addExpense("Coffee", 4.50, "Food");

tracker.addExpense("Lunch", 12.00, "Food");

tracker.addExpense("Bus Ticket", 2.50, "Transport");

tracker.addExpense("Movie Ticket", 15.00, "Entertainment");

tracker.viewExpensesByCategory("Food");

tracker.generateExpenseReport();

}

}

**Output:**

**java -cp /tmp/A3uKe3IL7x/ExpenseTracker**

**Expenses for category: Food**

**Description: Coffee, Amount: 4.5, Category: Food**

**Description: Lunch, Amount: 12.0, Category: Food**

**Expense Report:**

**Category: Entertainment, Total: 15.0**

**Category: Transport, Total: 2.5**

**Category: Food, Total: 16.5**

**Total Expenses: 34.0**

**=== Code Execution Successful ===**