SET 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)**

A.

```
package helloworld12;
 class Account{
	String name="ganesh";
	double amount=20000.00;
	void Create_account()
{
	System.out.println("Account Holdername:"+name);
	System.out.println("Inital amount:"+amount+"$");
}

void depositmoney(int accountnumber, double deposiamount)
{
	amount=amount+deposiamount;
	System.out.println("AccountNumber :"+accountnumber+"$");
	System.out.println("deposiamount:"+deposiamount+"$");
	System.out.println("total amount:"+amount+"$");


}
void withdraw(double withamount)
{
	System.out.println("withdraw amount:"+withamount+"$");
	System.out.println("Balance:"+(amount-withamount+"$"));
}
}
 public class main1{
            public static void main(String [] args) {
                        Account a=new Account();
                        a.Create_account();
                        a.depositmoney(123455,5000);
                        a.withdraw(1000);
 }}
```

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()**

A.

```
import java.util.ArrayList;
import java.util.HashMap;
import java.util.List;
import java.util.Map;

public class ExpenseTracker {
    private List<Expense> expenses;

    public ExpenseTracker() {
        this.expenses = new ArrayList<>();
    }

    public void addExpense(String description, double amount, String category) {
        if (amount < 0) {
            throw new IllegalArgumentException("Amount cannot be negative.");
        }
        Expense newExpense = new Expense(description, amount, category);
        expenses.add(newExpense);
    }

    public List<Expense> viewExpensesByCategory(String category) {
        List<Expense> expensesByCategory = new ArrayList<>();

        for (Expense expense : expenses) {
            if (expense.getCategory().equals(category)) {
                expensesByCategory.add(expense);
            }
        }

        return expensesByCategory;
    }

    public void generateExpenseReport() {
        if (expenses.isEmpty()) {
            System.out.println("No expenses to generate a report.");
            return;
        }

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

        for (Expense expense : expenses) {
            String category = expense.getCategory();
            double amount = expense.getAmount();

            categoryTotal.put(category, categoryTotal.getOrDefault(category, 0.0) + amount);
        }
        System.out.println("Expense Report:");
        for (Map.Entry<String, Double> entry : categoryTotal.entrySet()) {
            System.out.println(entry.getKey() + ": $" + entry.getValue());
        }
    }

    private class Expense {
        private String description;
        private double amount;
        private String category;

        public Expense(String description, double amount, String category) {
            this.description = description;
            this.amount = amount;
            this.category = category;
        }

        public String getCategory() {
            return category;
        }

        public double getAmount() {
            return amount;
        }
    }
}
```