Q Give all possible outputs of the two concurrent threads

```
public void thread1() {
    String one = "A";
    System.out.println(one);
    String one = "B";
    System.out.println("B");
}
```

```
public void thread2() {
    System.out.println("C");
    System.out.println("D");
}
```

2

Q1: Give all possible outputs of the two concurrent threads

Q Is the following code safe from concurrency issues?

```
private int balance = 150;
int getBalance() {
    return balance;
}
void withdraw(int amount) {
    if(amount > getBalance())
        throw new WithdrawTooLargeException();
    // maybe balance changed, so get the new balance
    setBalance(getBalance() - amount);
}
```

4

Q1: Is the following code safe from concurrency issues?

Q How many lock ops are needed to synchronize addIngredient?

```
private Stack<String> smoothie = new Stack<>();
public String addIngredient(String ingredient) {
    if (smoothie.size() >= MAX) throw new SmoothieSizeException();
    if (ingredient.equals("banana")) {
        return "I'm allergic, sorry.";
    } else {
        smoothie.push(ingredient);
        return "Ingredient added!";
    }
}
```

6

Q1: How many lock ops are needed to synchronize addIngredient?

Q What is the most significant problem with swipeCard?

```
private Password[] passwords = new Password[N];
private int diningDollars = 1000;
void swipeCard(int amount) {
    synchronized(passwords[i]) {
        if (amount % 5 == 0) passwords[i] = new Password();

        int currentBalance = this.diningDollars;
        this.diningDollars = currentBalance - amount;
    }
}
```

8

Q1: What is the most significant problem with swipeCard?