1.

Błąd:

liczba przepisów niezgodna z wymaganiami biznesowymi

```
private final int NUM_RECIPES = 4;
```

Poprawka:

```
private final int NUM_RECIPES = 3;
```

2.

Metoda:

```
public boolean deleteRecipe(Recipe r)
```

Błąd:

zbędne ponowne przypisanie tego samego elementu tablicy zamiast jego nadpisania

```
recipeArray[i] = recipeArray[i];
```

Poprawka:

```
recipeArray[i] = new Recipe();
```

3.

Metoda:

```
public boolean addInventory(int amtCoffee, int amtMilk, int amtSugar, int amtChocolate)
```

Błąd:

Niewłaściwe sprawdzenie warunku dotyczącego ilości cukru powodowało niepowodzenie dodawania zasobów do Inventory (sygnalizowane zwróceniem wartości false)

```
if (amtCoffee < 0 || amtMilk < 0 || amtSugar > 0 || amtChocolate < 0)
```

Poprawka:

```
if (amtCoffee < 0 || amtMilk < 0 || amtSugar < 0 || amtChocolate < 0)
```

4.

Metoda:

```
public boolean editRecipe(Recipe oldRecipe, Recipe newRecipe)
```

Błąd:

- metoda próbowała znaleźć i podmienić obiekt newRecipe zamiast oldRecipe
- metoda tworzyła nowy obiekt klasy Recipe, zamiast go podmienić

```
if (newRecipe.equals(recipeArray[i])) {
    recipeArray[i] = new Recipe();
    if (addRecipe(newRecipe)) {
        canEditRecipe = true;
    } else {
        //Unreachable line of code
        canEditRecipe = false;
    }
}
```

Poprawka:

```
if (oldRecipe.equals(recipeArray[i])) {
    recipeArray[i] = newRecipe;
    canEditRecipe = true;
}
```

5.

Metoda:

```
public int makeCoffee(Recipe r, int amtPaid)
```

Błąd:

Niewłaściwy sposób obliczenia ilości kawy w Inventory po zrobieniu napoju (ilość kawy jest dodawana, a powinna zostać odjęta.

```
inventory.setCoffee(inventory.getCoffee() + r.getAmtCoffee());
```

Poprawka:

```
inventory.setCoffee(inventory.getCoffee() - r.getAmtCoffee());
```