EXCEPTION AND PITFALL INSPECTION

Team 3: D-Encoders

Project's name: Ferrinventory

Inspection by team 2: BugBusters

Code version: 0.4

Date: Wednesday, june 21th, 2023

INDICATOR 1: Exceptions

Aspects to corroborate:

- **Problem:** Check with letters

  Consequence: The program stops running because it has an error.

  Evidence:

```
<----BIENVENIDO A FERRINVENTORY---->
Que deseas hacer ?
1. Agregar productos

2. Imprimir reporte de inventario

3. Salir

asa
Exception in thread "main" java.util.InputMismatchException
        at java.base/java.util.Scanner.throwFor(Scanner.java:947)
        at java.base/java.util.Scanner.next(Scanner.java:1602)
        at java.base/java.util.Scanner.nextInt(Scanner.java:2267)
        at java.base/java.util.Scanner.nextInt(Scanner.java:2221)
        at ec.edu.espe.ferrinventory.controller.Menu.printMenu(Menu.java:29)
        at ec.edu.espe.ferrinventory.view.Ferrinventory.main(Ferrinventory.java:22)
C:\Users\LENOVO\AppData\Local\NetBeans\Cache\18\executor-snippets\run.xml:111: The fo
C:\Users\LENOVO\AppData\Local\NetBeans\Cache\18\executor-snippets\run.xml:68: Java re
BUILD FAILED (total time: 9 seconds)
```

- **Problem:** Check with negative numbers.

  Consequence: The program doesn't perform any action.

Evidence:

```
run:
<----BIENVENIDO A FERRINVENTORY---->
Que deseas hacer ?
1. Agregar productos

2. Imprimir reporte de inventario

3. Salir

-2
BUILD SUCCESSFUL (total time: 3 seconds)
```

- **Problem:** Check large numbers

  Consequence: The program stops running because it has an error.

  Evidence:

```
run:
<----BIENVENIDO A FERRINVENTORY---->
Que deseas hacer ?
1. Agregar productos

2. Imprimir reporte de inventario

3. Salir

8009785654787
Exception in thread "main" java.util.InputMismatchException: For input string: "8009785654787"
        at java.base/java.util.Scanner.nextInt(Scanner.java:2273)
        at java.base/java.util.Scanner.nextInt(Scanner.java:2221)
        at ec.edu.espe.ferrinventory.controller.Menu.printMenu(Menu.java:29)
        at ec.edu.espe.ferrinventory.view.Ferrinventory.main(Ferrinventory.java:22)
C:\Users\LENOVO\AppData\Local\NetBeans\Cache\18\executor-snippets\run.xml:111: The following erro
C:\Users\LENOVO\AppData\Local\NetBeans\Cache\18\executor-snippets\run.xml:68: Java returned: 1
BUILD FAILED (total time: 6 seconds)
```

INDICATOR 2: Pitfalls

Aspects to corroborate:

- **Uncommunicative Names ("wh", "wh2" doesn't communicate that it represents)**

```
Object wh= parser.parse(new FileReader(fileName: route2));
Object wh2= parser.parse(new FileReader(fileName: routeWH1));

JSONObject pr=(JSONObject) wh;
JSONObject pr2=(JSONObject) wh2;
```

- **Inconsistent Names(te method name is fileReader and the name of the String is filerRoute)**

```
public void fileReader() throws FileNotFoundException, IOException, ParseException{
    String fileRoute="data\\WareHouse#1.json";
    JSONParser parser = new JSONParser();
```

- **Types Embedded in Names(variables are clean)**

```
public class Product {
    private String id;
    private String name;
    private String category;
    private String subCategory;
    private String brand;
    private float cost;
    private String price;
    private int stock;
```

- **Long Methods(the methods do the same but with some changes so there are unnecessary lines of code.)**

```java
public void saveWareHouse1(WareHouse wareHouse){
    Gson gson= new GsonBuilder().setPrettyPrinting().create();
    String gsonWareHouse=gson.toJson(src:wareHouse);
    System.out.println(x: gsonWareHouse);
    JSONParser parser= new JSONParser();
    try {
        String routeWH1="data\\WareHouse#1.json";
        String route2="data\\helper.json";

        FileWriter writer= new FileWriter(fileName: routeWH1,append: false);
        writer.write(str:gsonWareHouse);
        writer.close();

        Object wh= parser.parse(new FileReader(fileName: route2));
        Object wh2= parser.parse(new FileReader(fileName: routeWH1));

        JSONObject pr=(JSONObject) wh;
        JSONObject pr2=(JSONObject) wh2;

        JSONArray array=(JSONArray) pr.get(key:"Contenido");

        array.add(e: wh2);
        String array2=gson.toJson(src:wh);
        FileWriter writer2 = new FileWriter(fileName: route2,append: false);
        writer2.write(str:array2);
        writer.close();
```

```java
public void saveWareHouse2(WareHouse wareHouse){
    Gson gson= new GsonBuilder().setPrettyPrinting().create();
    String gsonWareHouse=gson.toJson(src:wareHouse);
    System.out.println(x: gsonWareHouse);
    JSONParser parser= new JSONParser();
    try {
        String routeWH1="data\\WareHouse#2.json";
        String route2="data\\helper.json";

        FileWriter writer= new FileWriter(fileName: routeWH1,append: false);
        writer.write(str:gsonWareHouse);
        writer.close();

        Object wh= parser.parse(new FileReader(fileName: route2));
        Object wh2= parser.parse(new FileReader(fileName: routeWH1));

        JSONObject pr=(JSONObject) wh;
        JSONObject pr2=(JSONObject) wh2;

        JSONArray array=(JSONArray) pr.get(key:"Contenido");

        array.add(e: wh2);
        String array2=gson.toJson(src:wh);
        FileWriter writer2 = new FileWriter(fileName: route2,append: false);
        writer2.write(str:array2);
        writer.close();
```

- **Duplicate Code(In the save class there are 4 methods in which the code is copied and small changes are added.)**

```java
public void fileReader() throws FileNotFoundException, IOException, ParseException{
    String fileRoute="data\\WareHouse#1.json";
    JSONParser parser = new JSONParser();

    try {
        Object wh=parser.parse(new FileReader(fileName: fileRoute));
        JSONObject pr=(JSONObject) wh;
        JSONArray array=(JSONArray) pr.get(key:"Contenido");

        for(int i =0; i<array.size(); i++){
            JSONObject singleProduct =(JSONObject) array.get(index: i);
            System.out.println(x: singleProduct);
        }
        System.out.println(x: "PRoductos en la Bodega #1");
    } catch (JSONException e) {
        System.out.println("Error"+e.getMessage());
    }
    Scanner sc=new Scanner(source: System.in);
        System.out.println(x: "Presione Enter para continuar");
        sc.nextLine();
}
```

```java
public void fileReader2() throws FileNotFoundException, IOException, ParseException{
    String fileRoute="data\\WareHouse#2.json";
    JSONParser parser = new JSONParser();

    try {
        Object wh=parser.parse(new FileReader(fileName: fileRoute));
        JSONObject pr=(JSONObject) wh;
        JSONArray array=(JSONArray) pr.get(key:"Contenido");

        for(int i =0; i<array.size(); i++){
            JSONObject singleProduct =(JSONObject) array.get(index: i);
            System.out.println(x: singleProduct);
        }
        System.out.println(x: "Productos en la Bodega #1");
    } catch (JSONException e) {
        System.out.println("Error"+e.getMessage());
    }
    Scanner sc=new Scanner(source: System.in);
        System.out.println(x: "Presione Enter para continuar");
        sc.nextLine();
}
```

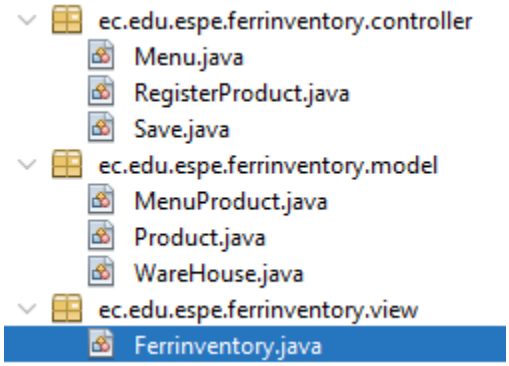
```java
public void saveWareHouse1(WareHouse wareHouse){
    Gson gson= new GsonBuilder().setPrettyPrinting().create();
    String gsonWareHouse=gson.toJson(src:wareHouse);
    System.out.println(x: gsonWareHouse);
    JSONParser parser= new JSONParser();
    try {
        String routeWH1="data\\WareHouse#1.json";
        String route2="data\\helper.json";

        FileWriter writer= new FileWriter(fileName: routeWH1,append: false);
        writer.write(str:gsonWareHouse);
        writer.close();

        Object wh= parser.parse(new FileReader(fileName: route2));
        Object wh2= parser.parse(new FileReader(fileName: routeWH1));

        JSONObject pr=(JSONObject) wh;
        JSONObject pr2=(JSONObject) wh2;

        JSONArray array=(JSONArray) pr.get(key:"Contenido");

        array.add(e: wh2);
        String array2=gson.toJson(src:wh);
        FileWriter writer2 = new FileWriter(fileName: route2,append: false);
        writer2.write(str:array2);
        writer.close();
```

```java
public void saveWareHouse2(WareHouse wareHouse){
    Gson gson= new GsonBuilder().setPrettyPrinting().create();
    String gsonWareHouse=gson.toJson(src:wareHouse);
    System.out.println(x: gsonWareHouse);
    JSONParser parser= new JSONParser();
    try {
        String routeWH1="data\\WareHouse#2.json";
        String route2="data\\helper.json";

        FileWriter writer= new FileWriter(fileName: routeWH1,append: false);
        writer.write(str:gsonWareHouse);
        writer.close();

        Object wh= parser.parse(new FileReader(fileName: route2));
        Object wh2= parser.parse(new FileReader(fileName: routeWH1));

        JSONObject pr=(JSONObject) wh;
        JSONObject pr2=(JSONObject) wh2;

        JSONArray array=(JSONArray) pr.get(key:"Contenido");

        array.add(e: wh2);
        String array2=gson.toJson(src:wh);
        FileWriter writer2 = new FileWriter(fileName: route2,append: false);
        writer2.write(str:array2);
        writer.close();
```

- **Long Message Chains(messages are short)**

```java
try {
    Object wh=parser.parse(new FileReader(fileName: fileRoute));
    JSONObject pr=(JSONObject) wh;
    JSONArray array=(JSONArray) pr.get(key:"Contenido");

    for(int i =0; i<array.size(); i++){
        JSONObject singleProduct =(JSONObject) array.get(index: i);
        System.out.println(x: singleProduct);
    }
    System.out.println(x: "PRoductos en la Bodega #1");
} catch (JSONException e) {
    System.out.println("Error"+e.getMessage());
}
Scanner sc=new Scanner(source: System.in);
    System.out.println(x: "Presione Enter para continuar");
    sc.nextLine();
```

- **Class Explosion (Doesn`t have subclasses that does the same thing)**



- **Large Message Chains (No more than 1 call, everything ok)**

```java
public class MenuProduct {
    public static void ProductMenu() {
        Product product;
        ArrayList<Product> products1=new ArrayList<>();
        ArrayList<Product> products2=new ArrayList<>();
        WareHouse wareHouse1 = new WareHouse(1, products1);
        WareHouse wareHouse2 = new WareHouse(2, products2);
        Menu printmenu=new Menu();
        RegisterProduct registerProduct= new RegisterProduct();
        Save saveInfo=new Save();

        boolean exit = true;

        while (exit==true){
            int option =printmenu.printProductMenu();
            switch(option){
                case 1:
                    System.out.println("-----Bodega#1 -----");
                    product=registerProduct.Product();
                    products1.add(product);
                    saveInfo.saveWareHouse1(wareHouse1);
                    break;
                case 2 :
                    System.out.println("-----Bodega#2 -----");
                    product=registerProduct.Product();
                    products2.add(product);
                    saveInfo.saveWareHouse2(wareHouse2);
                    break;
```

- **Large Classes (The project has max. 2 methods per class)**

```java
public class Menu {

    int option;
    public  int printProductMenu(){
        Scanner readOption = new Scanner(System.in);
        System.out.println("<----BIENVENIDO A FERRINVENTORY---->");
        System.out.println("Escoja la Bodega:");
        System.out.println("1. Bodega #1\n");
        System.out.println("2. Bodega #2\n");
        System.out.println("3. Salir\n");
        option = readOption.nextInt();
        return option;
    }
    public  int printMenu(){
        Scanner readOption = new Scanner(System.in);
        System.out.println("<----BIENVENIDO A FERRINVENTORY---->");
        System.out.println("Que deseas hacer ?");
        System.out.println("1. Agregar productos\n");
        System.out.println("2. Imprimir reporte de inventario\n");
        System.out.println("3. Salir\n");
        option = readOption.nextInt();
        return option;
    }
```

- **Conditional Complexity (Not hard to understand what the system is doing)**

```java
public class WareHouse {
    private int Id;
    private ArrayList<Product> products;

    @Override
    public String toString() {
        return "Bodega" + "#" + Id + "\nContenido=" + products + '\n';
    }



    public WareHouse(int Id, ArrayList<Product> products) {
        this.Id = Id;
        this.products = products;
    }
```

- **Oddball Solution (Maybe similar methods but not the same functionality)**

```java
public class Menu {

    int option;
    public  int printProductMenu(){
        Scanner readOption = new Scanner(System.in);
        System.out.println("<----BIENVENIDO A FERRINVENTORY---->");
        System.out.println("Escoja la Bodega:");
        System.out.println("1. Bodega #1\n");
        System.out.println("2. Bodega #2\n");
        System.out.println("3. Salir\n");
        option = readOption.nextInt();
        return option;
    }
    public  int printMenu(){
        Scanner readOption = new Scanner(System.in);
        System.out.println("<----BIENVENIDO A FERRINVENTORY---->");
        System.out.println("Que deseas hacer ?");
        System.out.println("1. Agregar productos\n");
        System.out.println("2. Imprimir reporte de inventario\n");
        System.out.println("3. Salir\n");
        option = readOption.nextInt();
        return option;
```

- **Redundant comments**

```java
case 3 :
//    saveInfo.saveInfo(wareHouse1, wareHouse2);
    exit = false;
    break;

case 2 :
    //TODO
    printInfo.fileReader();
    printInfo.fileReader2();
    break;
```

- **Dead Code (case 3 is not used even though it is printed in menu)**

Ferrinventory.java × | WareHouse.java × | Product.java × | MenuProduct.java × | Save.java ×

Source | History

```java
switch(option){
    case 1:
        MenuProduct.ProductMenu();
        break;
    case 2 :
        //TODO
        printInfo.fileReader();
        printInfo.fileReader2();
        break;
    default:
        exit=false;
        break;
}
```

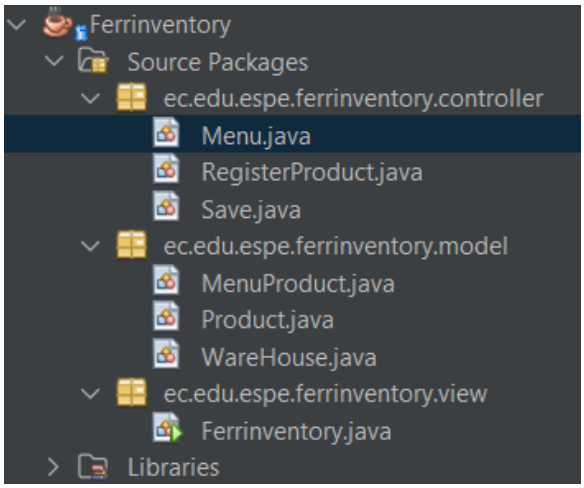
Output - Ferrinventory (run) ×

```
run:
<----BIENVENIDO A FERRINVENTORY---->
Que deseas hacer ?
1. Agregar productos

2. Imprimir reporte de inventario

3. Salir

4
BUILD SUCCESSFUL (total time: 1 minute 3 seconds)
```

(the exception isn't used)

```java
public void saveWareHouse2(WareHouse wareHouse){
    Gson gson= new GsonBuilder().setPrettyPrinting().create();
    String gsonWareHouse=gson.toJson(src:wareHouse);
    System.out.println(x: gsonWareHouse);
    JSONParser parser= new JSONParser();
    try {
        String routeWH1="data\\WareHouse#2.json";
        String route2="data\\helper.json";

        FileWriter writer= new FileWriter(fileName: routeWH1,append:false);
        writer.write(str:gsonWareHouse);
        writer.close();

        Object wh= parser.parse(new FileReader(fileName: route2));
        Object wh2= parser.parse(new FileReader(fileName: routeWH1));

        JSONObject pr=(JSONObject) wh;
        JSONObject pr2=(JSONObject) wh2;

        JSONArray array=(JSONArray) pr.get(key:"Contenido");

        array.add(e: wh2);
        String array2=gson.toJson(src:wh);
        FileWriter writer2 = new FileWriter(fileName: route2,append:false);
        writer2.write(str:array2);
        writer.close();

    } catch (Exception e) {
    }
}
```

- **Speculative Generality (menus could be in the same controller package, also, use the default domain of the university and not the team's domain)**



- **Temporal Field (ok, all variables are used)**

```java
*/
public class Product {
    private String id;
    private String name;
    private String category;
    private String subCategory;
    private String brand;
    private float cost;
    private String price;
    private int stock;

    public Product(String id, String name, String category, String subCategory, String brand, float cost, int stock) {
        this.id = id;
        this.name = name;
        this.category = category;
        this.subCategory = subCategory;
        this.brand = brand;
        this.cost = cost;
        this.stock = stock;
        this.price=calculatePrice();
    }
    private String calculatePrice() {
        float costFinal=(float) (cost/1.12);
        float priceOnSale=(float) (costFinal/0.75);
        float priceIva = (float) (priceOnSale*1.12);
        String PVP=String.format("%.2f", priceIva);
        return PVP;
        }

    @Override
    public String toString() {
        return "\n\t------------" + "\n\tNombre=" + name+ "\n\tCodigo=" + id  + "\n\tCategoria=" + category +
                "\n\tSubcategoria=" + subCategory + "\n\tMarca=" + brand +
                "\n\tCosto=" + cost + "\n\tPVP=" + price+ "\n\tstock=" + stock + '\n';
    }
```

- **Refused Bequest (menu class only print the message when printProductMenu method could be in the same MenuProduct class)**

```java
Menu printmenu=new Menu();
RegisterProduct registerProduct= new RegisterProduct();
Save saveInfo=new Save();

boolean exit = true;

while (exit==true){
    int option =printmenu.printProductMenu();
    switch(option){
        case 1:
            System.out.println("-----Bodega#1 -----");
            product=registerProduct.Product();
            products1.add(product);
            saveInfo.saveWareHouse1(wareHouse:wareHouse1);
            break;
        case 2 :
            System.out.println("-----Bodega#2 -----");
            product=registerProduct.Product();
```

```java
 * @author Moreno Paul,The encoders; DCCO-ESPE
 */
public class Menu {

    int option;
    public  int printProductMenu(){
        Scanner readOption = new Scanner(source:System.in);
        System.out.println("<----BIENVENIDO A FERRINVENTORY---->");
        System.out.println("Escoja la Bodega:");
        System.out.println("1. Bodega #1\n");
        System.out.println("2. Bodega #2\n");
        System.out.println("3. Salir\n");
        option = readOption.nextInt();
        return option;
    }
```

- **Inappropriate Intimacy (ok, internal fields and methods was encapsulated for only one class)**

```
private String id;
private String name;
private String category;
private String subCategory;
private String brand;
private float cost;
private String price;
private int stock;
```

```
private String calculatePrice() {
    float costFinal=(float) (cost/1.12);
    float priceOnSale=(float) (costFinal/0.75);
    float priceIva = (float) (priceOnSale*1.12);
    String PVP=String.format(format:"%.2f", args: priceIva);
    return PVP;
    }
```

- **Feature Envy (ok, the RegisterProduct class accesses the same number of data from the public Product method)**

```
public class RegisterProduct {
    public Product Product(){
        Scanner readProduct = new Scanner(source:System.in);

        System.out.print(s: "Nombre:");
        String name = readProduct.nextLine();
        System.out.print(s: "Id: ");
        String id = readProduct.nextLine();
        System.out.print(s: "Categoria: ");
        String category = readProduct.nextLine();
        System.out.print(s: "Sub Categoria: ");
        String subCategory=readProduct.nextLine();
        System.out.print(s: "Marca: ");
        String brand=readProduct.nextLine();
        System.out.print(s: "Stock:");
        int stock = readProduct.nextInt();
        while (stock<=-1){
            System.out.println(x: "No se permite stock negativo");
            System.out.print(s: "Stock:");
            stock = readProduct.nextInt();
        }
        System.out.print(s: "Costo de compra(usar coma):");
        float cost = readProduct.nextFloat();
        while(cost<=-1){
            System.out.println(x: "El costo no puede ser negativo");
            System.out.print(s: "Costo de compra:");
            cost = readProduct.nextFloat();
        }
        Product product=new Product(id, name, category, subCategory, brand, cost, stock);
        return product;
    }

}
```