PizzaStore - version 3

> Design Document by LeeKyeongJun

Index

1. What's New in PizzaStore version 3
2. PizzaStore.java
3. IFileHandler.java
4. CSVHandler.java
5. TXTHandler.java
6. MainClass.java
7. Test

What's New?

- 기존 `ShowErrMsg(int errcode)` 로 정의되었던 에러메시지 출력체계가 일관성이 없고, 코드의 가독성을 해쳐 해당 코드를 삭제하였습니다.
  - 에러 발생시 이제 해당하는 블럭에 바로 에러메시지를 출력하는 코드를 작성하였습니다.
- 에러메시지의 체계를 통일하였습니다.
  - 기존 에러메시지의 출력 형식이 통일되어있지 않아 이를 통일하였습니다.
    - 기존 프로그램의 재 실행이 가능한 에러의 경우 `"Error : Error Message"` 의 형식으로 통일하였습니다.
    - 기존 프로그램의 재 실행이 불가능한 심각한 에러의 경우 `"Fatal Error : Error Message"` 의 형식으로 통일한 뒤 프로그램을 종료하였습니다.
- CSV, TXT 파일로부터 PizzaStore의 정보를 읽고, 쓸 수 있도록 변경하였습니다.
- PizzaStore.java에서, `toString(), toString(int mod), toParsableString()` 함수를 추가 및 변경하였습니다.
  - 함수의 상세 구현은 아래를 참고하십시오.
- 버그 수정
  - Order를 만들고 난 뒤, 해당 Order를 변경했을 때 (`changeOrder`), 재 정산된 Order의 가격이 PizzaStore에 반영되지 않는 버그를 수정하였습니다.

PizzaStore.java

```
// in Line 20~
public PizzaStore(double cash, int peperoniStock, int mushroomStock, int cheeseStock) {
        this.cash = cash;
        this.peperoniStock = peperoniStock;
        this.mushroomStock = mushroomStock;
        this.cheeseStock = cheeseStock;
}

// in Line 136~
public String toString() {
                String ret = "PizzaStore : cash: $" + cash + ", peperoni:" + peperoniStock + ", mushrooms:" + mushroomStock+ ", cheeses:" + cheeseStock +".";
                return ret;
        }

public String toString(int mod) {
        if(mod == 0 && currentOrder != null) { // return Order
                return currentOrder.toString();
        }
        else return "";
}

public String toParsableString() {
        return cash + "/" + peperoniStock + "/" + mushroomStock + "/" + cheeseStock;
}
```

- PizzaStore의 생성자 추가
  - 인자를 받아 PizzaStore를 생성하는 함수를 추가하였습니다.
- toString() 함수의 세분화
  - 기존 `toString()` 의 경우, `currentOrder != null` 인 경우 `currentOrder.toString()` 을 return하고, 그렇지 않은 경우 PizzaStore의 정보를 return 하였습니다.
  - 새롭게 변경된 함수는 아래와 같습니다.
    - 기존 `toString()` 은 PizzaStore의 정보를 return 하는 것으로 한정합니다.
    - PizzaStore의 Instance Variable의 정보를 PizzaStore를 통해 return받고자 하면, 이제 toString의 Overloaded function인 `toString(int mod)` 를 사용합니다.
      - 초기값인 0의 경우 PizzaStore의 `currentOrder.toString()` 을 리턴합니다.
    - 파일에 작성하기 위해 PizzaStore의 정보를 넘겨주는 `toParsableString()` 함수가 추가되었습니다.

- 해당 함수는 PizzaStore의 cash, peperoniStock, mushroomStock, cheeseStock을 "/"를 구분자로 묶어 return 합니다.

IFileHandler.java

```
package Assignment1_code_Leekyeongjun_2019092824;

public interface IFileHandler {
	PizzaStore InitPizzaStore(String fileLocation);
	void writeToFile(String fileLocation, String text);
}
```

- IFileHandler는 CSVHandler와, TXTHandler의 Interface 입니다.

CSVHandler.java

```
package Assignment1_code_Leekyeongjun_2019092824;

import java.util.Scanner;
import java.io.FileInputStream;
import java.io.FileNotFoundException;
import java.io.PrintWriter;
import java.io.FileOutputStream;

public class CSVHandler implements IFileHandler{

	@Override
	public PizzaStore InitPizzaStore(String fileLocation) {
		// initial Setting
		if (!fileLocation.endsWith(".csv")) {
			throw new IllegalArgumentException("Error : This file is not a CSV file.");
		}

		Scanner inputStream = null;
		try {
			inputStream =
					new Scanner(new FileInputStream(fileLocation));
		}
		catch(FileNotFoundException e) {
			System.out.println("Error : Cannot found " + fileLocation);
			return null;
		}

		double money = 0;
		int mushroom = 0, peperoni = 0, cheese = 0;

		// Parsing CSV file
		while(inputStream.hasNextLine()) {
			try {
			String s = inputStream.nextLine();
				String line = s;
				String[] parts = line.split(",(?=(?:[^\"]*\"[^\"]*\")*[^\"]*$)", -1);
				if (parts.length != 2) {
					throw new IllegalArgumentException("Wrong Order format: " + line);
				}
				String key = parts[0].trim();
				String value = parts[1].trim().replaceAll("\"", "").replace(",", ".");

				switch (key) {
					case "Money":
						money = Double.parseDouble(value);
						break;
					case "Mushroom":
						mushroom = Integer.parseInt(value);
						break;
					case "Peperoni":
						peperoni = Integer.parseInt(value);
						break;
					case "Cheese":
						cheese = Integer.parseInt(value);
						break;
					default:
						throw new IllegalArgumentException("Unknown variable: " + key);
				}
		    }
			catch (IllegalArgumentException e) {
		    System.out.println("Error : " + e.getMessage());
```

```
                    return null;
                        }
                }
                return new PizzaStore(money, mushroom, peperoni, cheese);
        }

        @Override
    public void writeToFile(String fileLocation, String text) {

                if (!fileLocation.endsWith(".csv")) {
                        throw new IllegalArgumentException("Error : This file is not a CSV file.");
                }

        // Text format: "Money/Mushroom/Peperoni/Cheese", delimiter is "/"
        String[] parts = text.split("/");
        if (parts.length != 4) {
            throw new IllegalArgumentException("Error : The text format is incorrect.");
        }

        // Parse each part
        double money = Double.parseDouble(parts[0]);
        int mushroom = Integer.parseInt(parts[1]);
        int peperoni = Integer.parseInt(parts[2]);
        int cheese = Integer.parseInt(parts[3]);

        // Prepare the content in the desired format
        String content = String.format(
            "Money,\"%.2f\"\nMushroom,%d\nPeperoni,%d\nCheese,%d",
            money, // Format as a decimal number with comma as decimal separator
            mushroom,
            peperoni,
            cheese
        ).replace('.', ',');

        // Write to file
        try (PrintWriter writer = new PrintWriter(new FileOutputStream(fileLocation))) {
            writer.print(content);
        } catch (FileNotFoundException e) {
            e.printStackTrace();
            System.out.println("Fatal Error : An error occurred while trying to write to the file : " + fileLocation);
            System.exit(0);
        }
    }

}
```

InitPizzaStore

- InitPizzaStore함수는 CSV파일을 넘겨받아 이를 Parsing 하고, 해당 정보를 바탕으로 PizzaStore를 생성하여 return 합니다.
  - 만일 exception이 발견되었을 경우, 해당 함수는 null을 return 하고 종료합니다.

`// initial setting`

여기서는 주어진 파일이 `.csv` 형식으로 끝나지 않을 경우, `IllegalArgumentException` 을 throw합니다.
해당 exception은 `mainclass` 에서 받아, 잘못된 입력임을 알리고 다른 파일 주소를 입력하도록 유도합니다.
이후 실제로 해당 파일이 존재하는지 확인하고, 없을 경우 `FileNotFoundException` 을 catch하여 null을 return합니다.

`// Parsing csv file`

주어진 csv 파일을 한 줄씩 읽어, 정규식 `,(?=(?:[^\"]*\"[^\"]*\")*[^\"]*$)` 에 맞게 Parsing합니다.

- 정규식 `,(?=(?:[^\"]*\"[^\"]*\")*[^\"]*$)` 은 다음으로 구성되어 있습니다.
  - `,` : 쉼표를 찾습니다. 이것이 분할 기준입니다.
  - `(?= ... )` : 이것은 긍정형 전방 탐색입니다. 쉼표(,) 뒤에 오는 패턴을 확인하지만, 실제로 그 패턴을 결과에 포함시키지는 않습니다.
  - `(?: ... )` : 이것은 캡처하지 않는 그룹입니다. 패턴을 그룹화할 필요가 있지만, 그룹을 캡처하고 싶지 않을 때 사용합니다.
  - `[^\"]*` : 따옴표(`"`)가 아닌 문자가 0개 이상 연속되는 부분과 일치합니다. 여기서 `^` 는 부정을 의미하며, `\"` 는 이스케이프된 따옴표를 나타냅니다.
  - `\"[^\"]*\"` : 따옴표로 묶인 문자열과 일치합니다. 따옴표(`"`)로 시작해서 따옴표(`"`)로 끝나며, 그 사이에 따옴표가 아닌 문자들이 올 수 있습니다.
  - `*` : 앞의 패턴이 0번 이상 반복될 수 있음을 의미합니다.
  - `[^\"]*$` : 따옴표가 아닌 문자들이 0개 이상 반복되고 문자열의 끝(`$`)에 도달하는 부분과 일치합니다.

종합해보면, 이 정규식은 쉼표(,)를 찾되, 그 쉼표가 따옴표로 묶인 문자열 안에 있지 않고 문자열의 끝까지 따옴표가 아닌 문자들만 이어져 있을 경우에만 해당 쉼표를 기준으로 분할하겠다는 의미입니다. 이렇게 함으로써, 쉼표가 값의 일부로 포함된 경우 (예: `"15,00"`)에도 올바르게 처리할 수 있게 됩니다.

> 해당 정규식 작성 방법은 https://offbyone.tistory.com/400 및 Java 공식문서 https://docs.oracle.com/javase/7/docs/api/java/util/regex/Pattern.html 를 참고했습니다.

해당 Parsed String을 key와 value로 구분지어 Length가 2인 배열 Parts에 삽입한 후, `trim` 하여 공백을 제거, `replace` 를 이용해 double인 Money값이 올바르게 인식될 수 있도록 `,` 를 `.` 으로 바꿔주었습니다.

이후 Key에 따라 PizzaStore를 초기화 할 값을 설정했습니다. (아래의 Switch 구문 참조) 그리고 적절하지 않은 키워드 (예시에서는 Olive)가 등장할 경우, `IllegalArgumentException`을 throw하도록 하였습니다.

writeToFile

파일에 PizzaStore의 정보를 작성할 때, `"Money/Mushroom/Peperoni/Cheese"` 의 형식을 유지하도록 `toParsableString()` 함수를 구현해 두었습니다. 이를 바탕으로 `String.format`을 이용해
`"Money,\"%.2f\"\nMushroom,%d\nPeperoni,%d\nCheese,%d",` 형식으로 작성하도록 하였습니다.

> Money의 경우 제공된 예시 파일 (PizzaStore.csv)의 형식대로 `.` 를 `,` 로 치환, `""` 를 묶어 작성하도록 추가적인 수정을 진행했습니다.
> 아울러 원본에서 double이 소수점 2자리 수까지 작성되어있었으므로 여기서도 그 형식을 따랐습니다.

이후 해당 String을 제공된 파일에 덮어써 마무리하였습니다.

TXTHandler.java

```
package Assignment1_code_Leekyeongjun_2019092824;

import java.io.FileInputStream;
import java.io.FileNotFoundException;
import java.io.FileOutputStream;
import java.io.PrintWriter;
import java.util.Scanner;

public class TXTHandler implements IFileHandler{

        @Override
        public PizzaStore InitPizzaStore(String fileLocation) {

            if (!fileLocation.endsWith(".txt")) {
                throw new IllegalArgumentException("Error : This file is not a TXT file.");
            }

            Scanner inputStream = null;
            try {
                inputStream =
                        new Scanner(new FileInputStream(fileLocation));

            }
            catch(FileNotFoundException e) {
                System.out.println("Error : Cannot found " + fileLocation);
                return null;
            }

            double money = 0;
            int mushroom = 0, peperoni = 0, cheese = 0;

            while(inputStream.hasNextLine()) {
                try {
                    String s = inputStream.nextLine();
                    String line = s;
                    String[] parts = line.split(";", -1);
                    if (parts.length != 2) {
                        throw new IllegalArgumentException("Wrong Order format: " + line);
                    }
                    String key = parts[0].trim();
                    String value = parts[1].trim();

                    switch (key) {
                        case "Money":
                            money = Double.parseDouble(value);
                            break;
                        case "Mushroom":
                            mushroom = Integer.parseInt(value);
                            break;
                        case "Peperoni":
                            peperoni = Integer.parseInt(value);
                            break;
                        case "Cheese":
                            cheese = Integer.parseInt(value);
                            break;
                        default:
                            throw new IllegalArgumentException("Unknown variable: " + key);
                    }
                }
                catch (IllegalArgumentException e) {
                    System.out.println("Error : " + e.getMessage());
                    return null;
                }
```

```
            }
            return new PizzaStore(money, mushroom, peperoni, cheese);
        }

        @Override
        public void writeToFile(String fileLocation, String text) {

            if (!fileLocation.endsWith(".txt")) {
                throw new IllegalArgumentException("Error : This file is not a TXT file.");
            }

            // Text format: "Money/Mushroom/Peperoni/Cheese", delimiter is "/"
            String[] parts = text.split("/");
            if (parts.length != 4) {
                throw new IllegalArgumentException("Error : The text format is incorrect.");
            }

            // Parse each part
            double money = Double.parseDouble(parts[0]);
            int mushroom = Integer.parseInt(parts[1]);
            int peperoni = Integer.parseInt(parts[2]);
            int cheese = Integer.parseInt(parts[3]);

            // Prepare the content in the desired format
            String content = String.format(
                "Money;%.2f\nMushroom;%d\nPeperoni;%d\nCheese;%d",
                money,
                mushroom,
                peperoni,
                cheese
            );

            // Write to file
            try (PrintWriter writer = new PrintWriter(new FileOutputStream(fileLocation))) {
                writer.print(content);
            } catch (FileNotFoundException e) {
                e.printStackTrace();
                System.out.println("Fatal Error : An error occurred while trying to write to the file: " + fileLocation);
                System.exit(0);
            }
        }

}
```

TXTHandler의 경우 대부분을 CSVHandler와 공유하고 있습니다. 따라서, 다른 부분에 대해서만 설명하겠습니다.

- 파일의 확장자명이 .csv가 아닌 .txt 인지 검사합니다.
- delimeter가 ";"로, 이를 기준으로 값을 구분합니다.
- 파일의 작성시에도 원본 형식 `"Money;%.2f\nMushroom;%d\nPeperoni;%d\nCheese;%d"` 대로 작성합니다.

MainClass.java

```
// In line 6...
public void selectAction(PizzaStore store, Scanner input, Object FileHandler, String fileLocation) {

        int cmd = 0;
        System.out.println(store.toString());
        System.out.println("What would you like to do:");
        System.out.println("1: Place an order, 2: buy ingredients, 3: save and quit");

        cmd = input.nextInt();
        input.nextLine();

        while(!(cmd == 1 || cmd == 2 || cmd == 3)) {

                System.out.println("Error : Unavailable Command, Please Retry.");
                cmd = input.nextInt();
                input.nextLine();
        }

        if(cmd == 1) {
                placeOrder(store, input);
        }
        else if(cmd == 2) {
                buyIngredients(store, input);
        }
        else if(cmd == 3) {
                if(FileHandler.getClass() == CSVHandler.class) {
```

```
                        CSVHandler c = (CSVHandler) FileHandler;
                        c.writeToFile(fileLocation, store.toParsableString());
                        System.exit(0);
                }
                else if(FileHandler.getClass() == TXTHandler.class) {
                        TXTHandler t = (TXTHandler) FileHandler;
                        t.writeToFile(fileLocation, store.toParsableString());
                        System.exit(0);
                }
                else {
                        System.out.println("Fatal Error : There is no File Handler!");
                        System.exit(0);
                }
        }
}
```

- `public void selectAction(PizzaStore store, Scanner input, Object FileHandler, String fileLocation)` 의 경우 저장을 위해 FileHandler와, fileLocation을 인자로 받습니다.
    - Filehandler는 CSV, TXT Handler 둘중 하나를 upcasting 하여 Object class type으로 인자를 전달했고, 실제 Handler를 사용해야 하는 `else if(cmd == 3)` 에서는 클래스 이름의 직접 비교를 통해 안전한 상태에서 Downcasting 하여 사용했습니다.

```
        // in line 108...
        public void changeOrder(PizzaStore store, Scanner input) {
                boolean finished = false;
                while(finished == false) {
                        String cmd;
                        System.out.println("What do you want to do?");
                        System.out.println("1: Add a pizza, 2: Remove a pizza, 3: Nothing");
                        cmd = input.nextLine();

                        if(cmd.equals("1")) {
                            int size;
                            boolean hasPep= false, hasMus = false, hasChe = false;

                            System.out.println("What size pizza do you want?");

                            size = input.nextInt();

                            input.nextLine(); // flush nextLine after nextInt();
                            while(size < 0) {
                                System.out.println("Error : Unavailable Size, Please Retry.");
                                size = input.nextInt();
                                input.nextLine(); // flush nextLine after nextInt();
                            }

                            hasPep = selectIngredient(store, input, "peperoni");
                            hasMus = selectIngredient(store, input, "mushrooms");
                            hasChe = selectIngredient(store, input, "cheese");

                            store.AddPizzaToOrder(size, hasPep, hasMus, hasChe);
                        }
                        else if(cmd.equals("2")) {
                                System.out.println("Which pizza do you want to remove?");
                                int index;
                                index = input.nextInt();
                                input.nextLine();
                                store.removeCurrentOrder(index);
                        }
                        else if(cmd.equals("3")) {
                                finished = true;
                                System.out.println("Your final order is :");
                                System.out.println(store.toString(0));
                                store.addCash();
                        }
                        else {
                                System.out.println("Fatal Error : Unavailable Command.");
                                System.exit(0);
                        }
                }

        }
```

- `public void changeOrder(PizzaStore store, Scanner input)` 에서 이제 주문 변경이 마무리 된 이후 새로워진 Order를 정산하여 가격을 cash에 추가하는 코드를 작성하였습니다 - `else if(cmd.equals("3"))` 참고

```
// In line 207...
public static void main(String[] args) {
        Scanner scanner =  new Scanner(System.in);
        MainClass mainClassInstance = new MainClass();
```

```
        TXTHandler txthandler = new TXTHandler();
        CSVHandler csvhandler = new CSVHandler();
        Object handler = null;
        PizzaStore myStore = null;

        // Initial Settings
        System.out.println("What type of file do you want to use?");
        System.out.println("1 : CSV, 2 : TXT");

        int cmd = 0;
        cmd = scanner.nextInt();
        scanner.nextLine();

        if(cmd != 1 && cmd != 2) {
                System.out.println("Fatal Error :  Unavailable Command.");
                System.exit(0);
        }

        boolean InitSucceed = false;
        String location = "";

        while(InitSucceed == false) {
                System.out.println("What is the Location of the file?");
                location = scanner.nextLine();

                if(cmd == 1) {
                        try {
                                myStore = csvhandler.InitPizzaStore(location);
                        }
                        catch(IllegalArgumentException e) {
                                System.out.println(e.getMessage());
                        }
                        if(myStore != null) {
                                handler = (Object) csvhandler;
                                InitSucceed = true;
                        }
                }else {
                        try {
                                myStore = txthandler.InitPizzaStore(location);
                        }
                        catch(IllegalArgumentException e) {
                                System.out.println(e.getMessage());
                        }
                        if(myStore != null) {
                                handler = (Object) txthandler;
                                InitSucceed = true;
                        }
                }
        }
        // main Jobs
        while(true) {
                mainClassInstance.selectAction(myStore, scanner, handler, location);
        }
}
```

- `public static void main(String[] args)` 에서 작업에 들어가기에 앞서, 파일의 타입을 결정하도록 하였고, 이에 따라 PizzaStore를 initialize 하기를 시도하며, exception이 발생한 경우 이를 Handling 하도록 하였습니다.

Test

Main Method의 명세는 다음과 같다.

```
-On startup, first promt the user which type of file they want to use (txt/csv)
-After choosing the filetype, ask for the location of the initialization file
-Initialize the PizzaStore trough the chosen fileHandler
-Create 3 orders, the amount of pizza's and the ingredients are not relevant
-Update the initialization file with the store's current money and ingredients upon closing the program
```

Case : Normal CSV File

- Before Executing

```
Money,"15,00"
Mushroom,5
```

```
Peperoni,4
Cheese,3
```

- Executing

```
What type of file do you want to use?
1 : CSV, 2 : TXT
1
What is the Location of the file?
C:\Users\babyn\Documents\testfiles\PizzaStore.csv
PizzaStore : cash: $15.0, peperoni:5, mushrooms:4, cheeses:3.
What would you like to do:
1: Place an order, 2: buy ingredients, 3: save and quit
1
What type of order?
1: In store, 2: Online, 3: back
2
Chosen : Online
What is delivery address?
ITBT 202
What size pizza do you want?
32
Do you want peperoni on your pizza? Y/N
Y
Do you want mushrooms on your pizza? Y/N
N
Do you want cheese on your pizza? Y/N
Y
Do you want to order another Pizza? (Y/N)
N
Your final order is :
Online Order
Pizza[0] : [Size = 32 / Ingredients = Peperoni Cheese]
Total Price : 9.4
Address : ITBT 202

Do you want to change your order? (Y/N)
N
PizzaStore : cash: $24.4, peperoni:4, mushrooms:4, cheeses:2.
What would you like to do:
1: Place an order, 2: buy ingredients, 3: save and quit
1
What type of order?
1: In store, 2: Online, 3: back
2
Chosen : Online
What is delivery address?
ITBT 303
What size pizza do you want?
24
Do you want peperoni on your pizza? Y/N
N
Do you want mushrooms on your pizza? Y/N
N
Do you want cheese on your pizza? Y/N
Y
Do you want to order another Pizza? (Y/N)
N
Your final order is :
Online Order
Pizza[0] : [Size = 24 / Ingredients = Cheese]
Total Price : 8.4
Address : ITBT 303

Do you want to change your order? (Y/N)
N
PizzaStore : cash: $32.8, peperoni:4, mushrooms:4, cheeses:1.
What would you like to do:
1: Place an order, 2: buy ingredients, 3: save and quit
1
What type of order?
1: In store, 2: Online, 3: back
1
Chosen : In Store
What size pizza do you want?
32
Do you want peperoni on your pizza? Y/N
Y
Do you want mushrooms on your pizza? Y/N
N
Do you want cheese on your pizza? Y/N
Y
```

```
Do you want to order another Pizza? (Y/N)
Y
What size pizza do you want?
28
Do you want peperoni on your pizza? Y/N
N
Do you want mushrooms on your pizza? Y/N
N
Do you want cheese on your pizza? Y/N
Y
The order is Not Available.
Do you want to order another Pizza? (Y/N)
Y
What size pizza do you want?
28
Do you want peperoni on your pizza? Y/N
N
Do you want mushrooms on your pizza? Y/N
N
Do you want cheese on your pizza? Y/N
N
Do you want to order another Pizza? (Y/N)
N
Your final order is :
InStore Order
Pizza[0] : [Size = 32 / Ingredients = Peperoni Cheese]
Pizza[1] : [Size = 28 / Ingredients = ]
Total Price : 15.409999999999998
TableNumber : 1

Do you want to change your order? (Y/N)
Y
What do you want to do?
1: Add a pizza, 2: Remove a pizza, 3: Nothing
2
Which pizza do you want to remove?
1
Removed Pizza[1].
What do you want to do?
1: Add a pizza, 2: Remove a pizza, 3: Nothing
3
Your final order is :
InStore Order
Pizza[0] : [Size = 32 / Ingredients = Peperoni Cheese]
Total Price : 7.359999999999999
TableNumber : 1

PizzaStore : cash: $40.16, peperoni:3, mushrooms:4, cheeses:0.
What would you like to do:
1: Place an order, 2: buy ingredients, 3: save and quit
3
```

- After execution

```
Money,"40,16"
Mushroom,3
Peperoni,4
Cheese,0
```

Case : Wrong CSV File

```
What type of file do you want to use?
1 : CSV, 2 : TXT
1
What is the Location of the file?
C:\Users\babyn\Documents\testfiles\PizzaStore_Error.csv
Error : Unknown variable: Olive

What is the Location of the file?
C:\Users\babyn\Documents\testfile\There_is_No_such_file.csv
Error : Cannot found C:\Users\babyn\Documents\testfile\There_is_No_such_file.csv

What is the Location of the file?
C:\Users\babyn\Documents\testfiles\PizzaStore.txt
Error : This file is not a CSV file.

What is the Location of the file?
```