Philosophy of Design:

The purpose of my program is to read in storage names and capacities and construct a storage facility consisting of a freezer, pantry and fridge. Each storage location will store either a fruit, vegetable, meat or grain item (object). The program will decipher between these documents based on the foods storage temperature. The storage temperature along with other details of the item can be inputted by the user or read in from the file in the following lines from the storage capacity.

Structure of program:

The program consists of a super class named "FoodClass." This class stores the name, storage temperature and packaging of each item as they are the common fields in the 4 food categories. This class was defined as abstract to ensure that an object called Food of type FoodClass would not be created. Correspondingly, there are 4 sub-classes. These classes are "MeatClass", "GrainClass", "FruitClass" and "VegetablesClass." Each of these classes have properties that are unique to their classes but not common across all four classes so are instantiated as separate classes. All 4 classes inherit from the Food super class, so that the subclasses gain the properties of the Food super class. Each of these classes incorporate the features that were mentioned in both the lectures and workshops. The classes get passed values after the user inputs the values or the file is read in.

Furthermore, the 5th class is the storage class which is associated with FoodClass. Although it doesn't inherit from the FoodClass, the storage class declares an array of type FoodClass. This storage class has 3 rows, each representing one of the storage locations, however each storage location has a different number of columns. By creating the storage of type FoodClass, the program can store food items in one of the storage locations.

The 6th class is the "IOclass". This class is responsible for reading in the storage name, storage capacities and if the file contains any, the food items following the capacities. The IOclass' methods are static as they do not belong to the object, but rather the class itself.

The final class is the main class. This class contains several submodules which are called in the main. The main class contains all the menus for the program and displays them when required. Although, this class has other functions such as 'remove food' which is associated to the storage class as it can remove certain items from storage.

IFood

<<Interface>>I
IFood

+ today (Date)
+ isExpired (Boolean)

+ calcExpiry (Date today)
+ calcSpace (FoodClass food)

StorageClass

+ storageArray (2D Array of type FoodClass)
+ fridgeCount (Integer)
+ freezerCount (Integer)
+ pantryCount (Integer)

+ findExpired( )
+ removeItem(location, itemNumber)
+ addFood (MeatClass meat)
+ addFood(GrainClass grain)
+ addFood(FruitClass fruit)
+ addFood(VegetablesClass vegetables)

- FoodClass

+ name (String)

+ storageTemp (double)

+ packaging (String

MeatClass

+ cut (String)

+ weight (double)

+ usebyDate

+ calcExpiry (Date today)

GrainClass

+ type (String)

+ volume (double)

+ bestBeforeDate (Date)

+ calcExpiry (Date today)

FruitClass

+ type (String)

+ numberOfPieces (Integer)

+ usebyDate (Date)

+ calcExpiry (Date today)

VegetablesClass

+ weight (double)

+ bestBeforeDate (Date)

+ calcExpiry (Date today)

Main

+ menuNumber( )
+ switchFoodMenuOption(StorageClass storage)
+ validateFileName( )
+ foodMenu ( )
+ addFood(String fileName, int foodOption)
+ removeFood(StorageClass storage)
+ getMeatInput( )
+ getGrainInput( )
+ getFruitInput( )
+ getVegetablesInput( )
+ displayStorageContents(StorageClass storage)
+ getValidDate(Scanner sc)

IOclass

+ getNoOfLines(String filename)
+ readFile(String fileName)
+ processStorage(String line)
+ processCapacity(String line)
+writeOutStorage(StorageClass storageArray)
+ readFoodLines(BufferedReader bufRdr, StorageClas
+ processFood(String line)