# DATA DICTIONARY

Created by Jordan McElwee - Red

|  |  |  |  |
| --- | --- | --- | --- |
| **FIELD NAME** | **DATA TYPE** | **DESCRIPTION** | **EXAMPLE** |
| basketItems | **Global** String  *Arraylist* | Contains the names of each item added from two different array lists such as our bread and hot food | If lasagne is chosen then (index 0) = “**Lasagne**” |
| hotFoodArray | **Global** String  *Arraylist* | Containing all of the hot food items outlined in Appendix B: Hot Food Price List. | hotFoodArray.add(“**Burger**”) |
| hotFoodMeal | **Global** String  *Arraylist* | Strings containing the type of meal deals on offer which are displayed to the customer on the hot food selection screen. | **“+ 1 side £5.00”** |
| hotFoodPrice | **Global** Double  *Arraylist* | Prices of the hot food items. Items and prices with the same index will be matched with each other on the menu. | (index 0) = £**4.50** |
| login | **String** | Initally blank string which the customer / staff member must type in a login detail that must equal staff\_1 or staff\_2 in order to obtain admin status. | login = **“”**; |
| mealPrice | **Global** Double  *Arraylist* | Prices of the meal deals. In the if statement for a meal deal, the price of a meal will only be used if hotFoodMeal != \*Nothing available\* | mealPrice.add(**5.00)** |
| newPrice | Double | In the stockUpdateHot method. Used to substitute a new price value in the hotFoodPrice Arraylist. | newPrice (**2.50**)  *hotFoodPrice*.set**(***option*-1, newPrice (**2.50)** |
| note | **Global** String | An ever-changing variable that accepts string. Used to ask the customer if they want any more requirements for their order and be able to type in a sauce of their choosing. | *note* = key.nextLine**()**;  (“**Mayo and red sauce”**)  *notes*.add**(**Mayo and red sauce**)** |
| notes | **Global** String  *Arraylist* | Takes from the “note” variable and adds it into an array list so It can be displayed in the receipt | notes.get(0)  Console : Notes **Mayo and red sauce** |
| option | **Global** Integer | Used to select an option on the mainmenu and is then reset in hotFood and is used for selecting the type item and which curry meal deal as there are 2 “options”. Better to reset and reuse rather than initalise a different variable. | option (**3**) = lunchToGoMerged.home(**3**;  switch (option)  **case 3:**  hotFoodSelection();  Runs hot food ordering |
| priceBasket | **Global**  Double  Arraylist | Stores the prices from both sandwich and hotFoodPrice / mealPrice to print onto the receipt. | priceBasket.add(hotFoodPrice.get(0 (**4.50)** |
| qty | **Global** Integer  *Arraylist* | Stores the quantity from the variable “quantity” for printing onto the receipt. Matches with index of the item already chosen. | qty.add(**4**)  *Reciept printout*  Quantity  **4** |
| quantity | **Global** Integer | Quantity of the item ordered. In the *quantity* method, customer is asked to enter a quantity which then gets added to an arraylist. | *quantity =* ***2*** |
| sauce | **Global** String  Arraylist | Used in both sandwich and hot food. Sauces are called in the sandwich section and the user is asked to select and option between 1-5. Hot food displays the sauces in a table, which the customer can key in a string in the notes method. | “**Mayo**” |
| sides | **Global** String  Arraylist | Sides contains only the sides outlined in appendix A. The sides are added onto the entire hot food array and are used in its own side menu when getting a meal deal. | “**Chips**” |
| sideOption | Integer | Side option needed to be used in the meal deal as we need to keep a hold of the option chosen from the meal deal. sideOption functions the same as option, allowing the customer to choose a numbered item from the sidesMenu method. | sideOption = **1** (Beans) |
| sidesPrice | Double | Sides price contains only the prices outlined in appendix B. The prices also get added onto the entire hot food price. | **0.80** |
| staff | **Global** String | Staff equivalates to the login variable if the details entered also match staff\_1 or staff\_2. Staff is then used elsewhere when an admin element is required such as… | “**treety**” |
| staff\_1 | String | 1st login credential. Set as a check for comparing the value of login to staff\_1. Used to stop unwanted entry into the admin part of the system, where items can be removed and updated. | **“jmcelwee”** |
| staff\_2 | String | 2nd login credential. Same as staff\_2, but with a different credential. In real life, a system would have more than one login detail. | “**treety**” |
| subTotal | **Global** Double Arraylist | Used in the qtyTotal method to assign subtotal the value of quantity \* itemPrice (the price of the item called upon in the parameter) qtyTotal = 2 \* 4.50 | qtyTotal = 2 \* 4.50  subtotal = **9.00** |
| sum | Double | In the totalPrice method. Used as a parameter, to calculate the sum of the list of subtotal array elements. | **50.00** |
| total | **Global**  Double | Substitutes the value from the variable “sum” in totalPrice method. Each subtotal element is added onto the totalPrice and gives a total. | **12.50** |
| yesNo | **Global** Char | Used in many cases in both sandwich and hot food. In hot food, its used to ask if the customer wants a meal deal and used to ask if customer wants any requirements or sauce. | **‘Y’** |

Created by Treety F – Black

|  |  |  |  |
| --- | --- | --- | --- |
| **FIELD NAME** | **DATA TYPE** | **DESCRIPTION** | **EXAMPLE** |
| **Treety’s Variables** |  |  |  |
| nSand | int | Counts the number of sandwiches ordered | nSand++ |
| nCont | int | Counts the number of contents in a sandwich | nCont++ |
| nExtras | int | Counts the number of extra items ordered | nExtras++ |
| breadChoice | int | It is used to choose a bread type | BreadChoice = 1 |
| spreadChoice | int | It is used to chose a spread type | spreadChoice = 2 |
| meatChoice | int | It is used to chose a meat type | meatChoice = *7* |
| SaladChoice | int | It is used to chose a salad type | saladChoice =  *8* |
| sauceChoice | int | It is used to chose a sauce type | sauceChoice = *4* |
| checkOut | char | Y/N option to go to different  part of programme | Y/N |
| extrasM | char | Y/N option to choose the extra meat | extrasM = Y/N |
| extrasS | char | Y/N option to choose the extra salad | extrasS = Y/N |
| anySalad | char | Y/N option to choose | anySalad = Y/N |
| sandwichPriceBasket | double | To add up the prices | sandwichPriceBasket += 0.35; |
| sandwichContent | ArrayList  String | To add to the array list of sandwich content and print later | sandwichContent.add(sauce.get(sauceChoice - 1) |
| sandwichExtraContent | ArrayList  String | To add to the array list of sandwich Extra content and print later | sandwichExtraContent.get(k), sandwichExtraPriceBasket.get(k) |
| bread | ArrayList  String | To print and then recall the bread type which has been added | sandwichContent.add(bread.get(breadChoice - 1)); |
| spread | ArrayList  String | To print and then recall the spread type which has been added what has been added | spread.get(spreadChoice - 1) |
| meat | ArrayList  String | To print and then recall the meat type which has been added | meat.get(0), meat.get(5) |
| salad | ArrayList  String | To print and then recall the salad type which has been added | salad.get(saladChoice - 1)); |
| sauce | ArrayList  String | To print and then recall the sauce type which has been added | sandwichContent.add(sauce.get(sauceChoice - 1)); |
| sandwichExtraPriceBasket | ArrayList  Double | To add and then recall the sandwich extras which has been chosen | sandwichContent.add(sauce.get(sauceChoice - 1)); |