**Pseudocode**

**Start Program**

**Start Class Main**

Import order

Import startend

Import delivery

Insert previous constructors

Construct wlc = Welcome

Construct menu = Menu

Construct select = FoodSelect

Construct totPrice = CalcTotal

Construct cardPay = Payment

Construct userAddress = Address

Construct deliverDate = DatePicker

Output wlc introduction

Output wlc checkTime (check user’s time)

Output “”

Output menuItems

Output “”

Output display (instructions for ordering food)

Output userSelect (allows user to order food)

Output “”

Output displayTax (displays label “total”)

Output plusTax (displays price plus tax)

Output “”

Output user charge (to input user’s card info)

Output chargeDisplay(display user’s card info)

Output “”

Output deliverAddress (to input the delivery address)

Output “”

Output deliDateInstruct (instructions for picking a date)

Output userDate (pop-up for the date)

Output “”

**End Class**

**Start Package Delivery**

**Start Interface GetAddress**

Initialize houseNums

Initialize ex1

String StreetName

String ex2

String city

String ex3

**End Interface**

**Start Class Delivery Limits**

Import util scanner

String “Ca”

String”United States”

**End Class**

**Start Class Address Details**

Address Details Inherits Delivery Limits

Implements GetAddress interface

Read

Reinitialize houseNums

Output “Enter House Number: “ + (“ex1 “)

Initialize num

Num = read numbers

return num

Restring StreetName

Output “Enter street name: “ + (“ex2”)

Read

String name

Name = read words

Return name

Restring City

Output “Enter City: “ + (“ex3”)

Read

String city

Name = read words

Return city

**End Class**

**Start Class Address**

Address Inherits AddressDetails

Output “\nInsert Delivery Address: ”

Initialize houseNumber

Output houseNums (12345)

String StreetName

Output “(”Main Street”)”

String City

Output “(“Moreno Valley”)”

String address

Output houseNumber + “” + StreetName + “,” + city + “,”

String limits

Output Ca + “,“+ United States

Output “Delivery Address: “ + address + limits

**End Class**

**Start Class Date Picker**

Import Awt Dimension

Import Awt Flowlayout

Import Awt ActionEvent

Import Awt ActionListener

Import JButton

Import JFrame

Import JLabel

Import JTextField

Import Startend.ThankYou

Import Startend.Welcome

Construct ty = ThankYou

Construct deliverDateCheck = Welcome

String date  
 String display

If display not there then

Allow swing to display by setting the default display to 0

Endif

Date pick instructions

Output “Delivery Date: ”

Output “- Only Press the ‘submit’ button when you are done entering the desired date.”

Pop-up instructions

Construct frame

Exit on when you close it

Size: 250,150

Construct label

Output “Enter date (MM/DD/YYYY): ”

Construct dateField

Size 10

Construct submitButton

Output “Submit”

Add listener to submitButton

Read date value

Close Pop-up

Output “You delivery date is: “ + date

Determine if order goes through (based on opening time)

Output ty method

Add label to the frame

Add dateField to the frame

Add submitButton to the frame

Make the frame visible

**End Class**

**Start interface getCardNums**

Long first four card numbers

Long second four card numbers

Long third four card numbers

Long last four card numbers

**End Interface**

**Start interface getCardType**

String type

**End Interface**

**Start Class Card Details**

Import util scanner

Import order.Calctotal

Implements getCardNums interface

Implements getCardType

Read

Initialize cardRead

String cardAllNums

String cardType

String type

Initialize cardOption

cardOption = read numbers

Case cardOption

Case 1:

cardType is “visa”

break

Case 2:

cardType is “Mastercard”

Break

Case 3:

cardType is “American Express”

break

Case 4:

cardType is “Discover”

break

Default:

cardType is “Invalid option. Please try again.”

break

End Case

Return cardType

Long first four card numbers

Initialize cardRead = read numbers

Return cardRead

Long second four card numbers

Initialize cardRead = read numbers

Return cardRead

Long third four card numbers

Initialize cardRead = read numbers

Return cardRead

Long last four card numbers

Initialize cardRead = read numbers

Return cardRead

**End Class**

**Start Class Payment**

Payment Inherits card details

Construct numDetails

Construct typeDetails

String userCardType

userCharge instructions

Output “n\Payment”

Output “”

Output “Card Type”

Output “-Enter the number next to the desired option to select the card type”

Output “Allowed payment types: “

Output “1. Visa\n2. Mastercard\n3. American Express/n4. Discover”

Output “Enter card type here: “

userCardType = user input

Output “”

Output “Card Number”

Output “- Enter the first set of numbers then enter the second set of four numbers, then

enter the third set of four numbers then enter the last set of four numbers” \n- Each set of

four on different lines to correctly input the card number”

Output “Enter card number here: “

Long Card1 = user input (first four numbers)

Long Card2 = user input (second four numbers)

Long Card3 = user input (third four numbers)

Long Card4 = user input (last four numbers)

cardAllNums = card1 + “-” + card2 + “-” + card3 + “-” + card4

Construct totPrice based on user’s food order

Add tax to the user’s total price

Output “ charged to” + userCardType + “” + cardAllNums

**End Class**

**End Package**

**Start Package Order**

**Start Class Menu**

Declare a two dimensional array {food item, price}

Output “Menu: ”

Set rows a

For 8 rows

Set columns b

For 2 rows

Endfor

Endfor

Output Array [a][b]

Output “”

**End Class**

**Start Class Food Select**

Import util scanner

Import util arraylist

Import List

Read

Declare a one dimensional array list {total price}

Declare a one dimensional array list {selected orders}

Output “Food order: “

Output “- Please type in the food item name exact to that of the 'Menu' followed by ENTER.”

Output “- Enter 'Complete Order' when you have completed your food order."

Output “Ex. \n\tCheeseburger\n\tHot Dog\n\tComplete Order"

Output “Enter Food Order Here: \n”

Declare a one dimensional array {all prices}

String constructor name food for each price in the array {all prices}

Set each object to a specific food item

While true

Read

If input reads “Checkout” then

Break

Endif

Endwhile

Input food item

Case food item

Cheeseburger:

add to selected orders & add it’s String constructor to totalprice

Hotdog:

add to selected orders & add it’s String constructor to totalprice

French Fries:

add to selected orders & add it’s String constructor to totalprice

Ground Beef Burrito:

add to selected orders & add it’s String constructor to totalprice

Chicken Taco:

add to selected orders & add it’s String constructor to totalprice

12” Pepperoni Pizza:

add to selected orders & add it’s String constructor to totalprice

10 pieces of Southern Fried Jumbo Shrimp:

add to selected orders & add it’s String constructor to totalprice

Fish and Chips:

add to selected orders & add it’s String constructor to totalprice

EndCase

If Input != Selected Orders then

Output ”Invalid item Please try again”

EndIf

Output “\nFood Summary

Output “- Includes 7.25% Tax”

Output “n\Food Item(s) ordered: “

Set priceperitem

For Output order + “- $” + totalprice

Bump priceperitem

EndFor

Output total price

**End Class**

**Start Class: Values**

Import util arraylist

Set tax to .0725

Set grantotal

Declare a one dimensional array {realPrice}

For t = every realPrice value

t bumps to grantotal

Output grantotal

Endfor

**End Class**

**Start Class: Calc Total**

Calctotal inherits values

Output “Total: “

Construct grantotal

Declare a one dimensional array {realPrice double}

For String grantotal to return total price

Use realPrice array list to convert old selected prices array to be double   
Endfor

Output “$” + total + tax

**End Class**

**End Package**

**Start Package Startend**

**Start Class ThankYou**

Output ””

Output “Thank you for your order!”

Output “Have a great day! :)”

Exit System

**End Class**

**Start Class Usertime**

Import delivery.DatePicker

Import time.localtime

Import time.zoneid

Import time.format

Set currentTime = local time zone of Los Angeles

Set Formatter = display time as hour:minute

String formattedTime

formattedTime = currentTime

Initialize am = morning limit between 11-0

Initialize pm = evening limit between 22-0

**End Class**

**Start Class Welcome**

Welcome inherits Usertime

Introduction

Output “”

Output “\”REGULAR RESTAURANT\””

Output “Business hours:\nMonday - Sunday: 11am - 10 pm”

Output “Welcome!”

Output “Current time:” + formattedTime

Check the time

If the current time is before 10 pm and after 11 am then

Output “We are open!”

Output “Place an order to go:”

Else

Output “Sorry we are closed”

Output “You may still place your order. However it will not be delivered until opening time”

Endif

Order Recap/Status

Construct appointment

If the current time is before 10 pm and after 11 am then

Output “Your order was successful!”

Else

Output “Your order will be placed when we open again. Thank you!”

**End Class**

**End Package**

**End Program**