|  |  |  |
| --- | --- | --- |
| Input | Process | output |
| int taxiType  double distance  int numStops | Int numCustomers  Algorithm   1. Call displayMenu 2. Ask for number of customers   numCustomers   1. Create for loop for numCustomers   For (int i = 0, I < numCustomers, i++)   1. Display taxi type menu 2. Ask user for taxi type, validate   taxiType   1. Ask user for distance   Distance   1. Ask user for number of stops   numStops   1. Call calcPerMileFee 2. Call calcBaseFee 3. Call calcTotalFee 4. Display totalFee 5. Call displayTaxiFee | Taxi type  Distance  numStops  taxiFee |

displayMenu

|  |  |  |
| --- | --- | --- |
| Input | Process | output |
|  |  | menu |

calcPerMileFee

|  |  |  |
| --- | --- | --- |
| Input | Process | output |
| int taxiType | final double STANDARD\_TAXI  final double BUSINESS\_TAXI  final double LUXURY\_TAXI  int taxiType  algorithm   1. Create switch   Switch(type) {  Case 1:  perMileFee = 1.50  break;  Case 2:  perMIleFee = 2.00  break;  default:  perMIleFee = 2.50  break  }   1. Return perMileFee | perMIleFee |

calcBaseFee

|  |  |  |
| --- | --- | --- |
| Input | Process | output |
| Double distance  Double perMileFee | Algorithm   1. baseFee = distance \* perMileFee 2. return baseFee | baseFee |

calcTotalFee

|  |  |  |
| --- | --- | --- |
| Input | Process | output |
| Double baseFee  Int numStops | Algorithm   1. totalFee = baseFee + (5.00 \* numStops) 2. return totalFee | totalFee |

displayTaxiFee

|  |  |  |
| --- | --- | --- |
| Input | Process | output |
| Double totalFee | Algorithm   1. aggregate totalFee |  |