SourceCode:

*"""  
ITMD 513 : Classes - HomeWork - Bhanu Kureti  
Summary : In this program Checkout\_Register will implement a class that provides same type functionality  
of real life checkout registers  
"""*class Checkout\_Register:  
 currencysymbol = ""  
 salesTax = 0  
 total\_transactions = 0  
 total\_amt\_for\_transactions = 0  
 num\_of\_cash\_transactions = 0  
 total\_amt\_for\_cash\_transactions = 0  
 num\_of\_card\_transactions = 0  
 total\_amt\_for\_card\_transactions = 0  
 productList = {}  
  
 def \_\_init\_\_(self, symbol, taxes):  
 *"""  
 Constructor for checkout register class  
 parameters:  
 symbol: The symbol of currency.  
 taxes: The sales tax for the transaction.  
 """* self.currencysymbol = symbol  
 self.salesTax = taxes  
  
 # Accessor methods of checkout register class  
 def getCurrencySymbol(self):  
 return self.currencysymbol # Returns Currency symbol  
  
 def getSalesTax(self):  
 return self.salesTax # Returns sales tax of transaction  
  
 def getTotalTransactionsNumber(self):  
 return self.total\_transactions # Returns total number of transactions  
  
 def getTotalTransactionsAmount(self):  
 return self.total\_amt\_for\_transactions # Returns the total amount for whole transactions  
  
 def getTotalCashTransactionsNum(self):  
 return self.num\_of\_cash\_transactions # Returns the total number of cash transactions  
  
 def getTotalCashTransactionsAmount(self):  
 return self.total\_amt\_for\_cash\_transactions # Returns the total amount for cash transactions  
  
 def getTotalCardTransactionsNum(self):  
 return self.num\_of\_card\_transactions # Returns the total number of card transactions  
  
 def getTotalCardTransactionsAmount(self):  
 return self.total\_amt\_for\_card\_transactions # Returns the total amount for card transactions  
  
 def getProductList(self):  
 return self.productList # returns the products list  
  
 # Setter methods of checkout register class  
  
 def setProductList(self, items1):  
 self.productList = items1 # set the products list  
  
 def setCurrencySymbol(self, symbol):  
 self.currencysymbol = symbol # set Currency symbol  
  
 def setSalesTax(self, tax):  
 self.salesTax = tax # set tax of transaction  
  
 def setTotalTransactionsNumber(self, totalTrans):  
 self.total\_transactions = totalTrans # set total number of transactions  
  
 def setTotalTransactionsAmount(self, totaltransAmt):  
 self.total\_amt\_for\_transactions = totaltransAmt # set the total amount for whole transactions  
  
 def setTotalCashTransactionsNum(self, totalCashTrans):  
 self.num\_of\_cash\_transactions = totalCashTrans # set the total number of cash transactions  
  
 def setTotalCashTransactionsAmount(self, totalCashAmt):  
 self.total\_amt\_for\_cash\_transactions = totalCashAmt # set the total amount for cash transactions  
  
 def setTotalCardTransactionsNum(self, totalCardTrans):  
 self.num\_of\_card\_transactions = totalCardTrans # set the total number of card transactions  
  
 def setTotalCardTransactionsAmount(self, totalCardAmt):  
 self.total\_amt\_for\_card\_transactions = totalCardAmt # set the total amount for card transactions  
  
 # This method will calculate the total amount of transaction  
 def calculateTotalAmount(self):  
 total = 0  
 #print(self.productList)  
 for i in self.productList:  
 total += self.productList[i]  
 return total  
  
 # This method will calculate tax on total transaction  
 def calculateSalesTax\_on\_transactions(self):  
 return self.salesTax \* self.calculateTotalAmount() / 100.0  
  
 # This method will return total amount with adding tax to it  
 def calculateTotalAmountWithTax(self):  
 return self.calculateTotalAmount() + self.calculateSalesTax\_on\_transactions()  
  
 # This method will return string which is receipt of the transaction  
 def createReceipt(self):  
 itemsList = "Items in your Transaction: \n"  
 total = 0  
 # Returns the list of items purchased  
 for i in self.productList:  
 itemsList += str(i) + " - " + self.currencysymbol + " " + str(self.productList[i]) + "\n"  
 total += self.productList[i]  
 #print(itemsList)  
 tax = self.calculateSalesTax\_on\_transactions()  
 itemsList += "--------------------\nSalesTax - " + self.currencysymbol + str(tax) + "\n"  
 total += tax  
 itemsList += "\n" + "Total Price -" + self.currencysymbol + " " + str(total)  
 return itemsList  
  
 # This method will process cash transaction  
  
 def processCash\_transaction(self, cash):  
 cash = float(cash)  
 self.total\_amt\_for\_transactions += cash  
 self.total\_transactions += 1  
 self.total\_amt\_for\_cash\_transactions += cash  
 self.num\_of\_cash\_transactions += 1  
 AmountDue = self.calculateTotalAmountWithTax()  
 # If cash given by customer is more than Due amount, remaining change is given to the customer  
 if cash > AmountDue:  
 print(self.createReceipt())  
 change = cash - AmountDue  
 return change  
 # If cash given by customer is less than Due amount, will returns the money owed by customer  
 elif cash < AmountDue:  
 AmountOwe = AmountDue - cash  
 print("Customer need to pay the due amount which is : "+str(AmountOwe))  
 exit()  
 return AmountOwe  
 else:  
 return 0  
  
 # This method will process card transactions  
  
 def processCard\_transaction(self, card\_type):  
 # if card type is other than debit or credit print invalid message  
 if (card\_type == "credit" or card\_type == "debit"):  
 AmountDue = self.calculateTotalAmountWithTax() # proceed with credit/debit card do the transactions  
 self.total\_amt\_for\_transactions += AmountDue  
 self.total\_transactions += 1  
 self.total\_amt\_for\_card\_transactions += AmountDue  
 self.num\_of\_card\_transactions += 1  
 # return the amount charged and print receipt  
 print(self.createReceipt())  
 return AmountDue  
 else:  
 print("\n Invalid card type provided, please provide either a credit or debit card")  
 return None

Outputs:

1. Debit card Transaction with values given

def main():  
 checkout = Checkout\_Register("$", 5)  
  
 items = {"pens": 10, "books": 200, "Bag": 500}  
 checkout.setProductList(items)  
  
 charged = checkout.processCard\_transaction("debit")  
 print("\nYour card has been charged -"+str(checkout.currencysymbol)+" "+str(charged))  
  
main()

Items in your Transaction:

pens - $ 10

books - $ 200

Bag - $ 500

--------------------

Total - $710

Sales Tax - $35.5

Total Price -$ 745.5

Your card has been charged -$ 745.5

Process finished with exit code 0

1. Card Transaction with invalid card type

def main():  
 checkout = Checkout\_Register("$", 5)  
  
 items = {"pens": 10, "books": 200, "Bag": 500}  
 checkout.setProductList(items)  
  
 charged = checkout.processCard\_transaction("bhanu")  
 print("\nYour card has been charged -"+str(checkout.currencysymbol)+" "+str(charged))  
  
main()

Invalid card type provided, please provide either a credit or debit card

Your card has been charged -$ None

Process finished with exit code 0

1. Cash Transaction with exact amount given by the customer

def main():  
 checkout = Checkout\_Register("$", 5)  
  
 items = {"pens": 10, "books": 200, "Bag": 500}  
 checkout.setProductList(items)  
  
 charged = checkout.processCash\_transaction(745.5)  
 #print("\nYour card has been charged -"+str(checkout.currencysymbol)+" "+str(charged))  
  
main()

Successfully transaction completed!!

Thanks for visiting:)

Process finished with exit code 0

1. Cash Transaction with amount given by customer is less than due amount

def main():  
 checkout = Checkout\_Register("$", 5)  
  
 items = {"pens": 10, "books": 200, "Bag": 500}  
 checkout.setProductList(items)  
  
 charged = checkout.processCash\_transaction(700)  
 #print("\nYour card has been charged -"+str(checkout.currencysymbol)+" "+str(charged))  
  
main()

Customer need to pay the due amount which is : 45.5

Process finished with exit code 0

1. If Customer gives more cash than due amount

def main():  
 checkout = Checkout\_Register("$", 5)  
  
 items = {"pens": 10, "books": 200, "Bag": 500}  
 checkout.setProductList(items)  
  
 charged = checkout.processCash\_transaction(800)  
 #print("\nYour card has been charged -"+str(checkout.currencysymbol)+" "+str(charged))  
  
main()

Items in your Transaction:

pens - $ 10

books - $ 200

Bag - $ 500

--------------------

Total - $710

Sales Tax - $35.5

Total Price -$ 745.5

Please give the customer remain change : 54.5

Process finished with exit code 0