หน้า Receipt.py

น.ส. ธฬพรรณ มาสถิตทรัพย์
61070507218
RC

Receipt.py

```python
from helper_functions import *
from Product import *
from Customer import *

class Receipt:
    def __init__(self):
        self.dict = {}

    def __updateLineItem (self, receiptLineItemList):
        receiptItemList = []
        total = 0
        for lineItem in receiptLineItemList:
            receiptLineItemDict = {}
            receiptLineItemDict["Invoice No"] = lineItem["Invoice No"]
            receiptLineItemDict["Amount Paid Here"] = lineItem["Amount Paid Here"]
            total += lineItem["Amount Paid Here"]
            receiptItemList.append(receiptLineItemDict)
        return receiptItemList, total

    def create(self, receiptNo, receiptDate, customerCode, paymentMethod, paymentReference, remark, receiptLineItemList):
        if receiptNo in self.dict:
            return {'Is Error': True, 'Error Message': "Receipt No '{}' already exists. Cannot Create. ".format(receiptNo)}
        else:
            receiptLineItemList,total = self.__updateLineItem(receiptLineItemList)

            self.dict[receiptNo] = {"Date" : receiptDate,"Customer Code" : customerCode,"Payment Method" : paymentMethod, "Payment Reference" : pay
        return {'Is Error': False, 'Error Message': ""}

    def read(self, receiptNo):
        if receiptNo in self.dict:
            retreceipt = self.dict[receiptNo]
        else:
            return ({'Is Error': True, 'Error Message': "Receipt No '{}' not found. Cannot Read.".format(receiptNo)},{})

        return ({'Is Error': False, 'Error Message': ""},retreceipt)

    def update(self, receiptNo, newReceiptDate, newCustomerCode, newPaymentMethod, newPaymentReference, newRemark, newReceiptLineItemList):
        if receiptNo in self.dict:
            self.dict[receiptNo]["Date"] = newReceiptDate
            self.dict[receiptNo]["Customer Code"] = newCustomerCode
            self.dict[receiptNo]["Payment Method"] = newPaymentMethod
            self.dict[receiptNo]["Payment Reference"] = newPaymentReference
            self.dict[receiptNo]["Remarks"] = newRemark
            receiptLineItemList,total = self.__updateLineItem(newReceiptLineItemList)

            self.dict[receiptNo]["Total Received"] = total
            self.dict[receiptNo]["Receipt Line Item"] = newReceiptLineItemList
        else:
            return {'Is Error': True, 'Error Message': "Receipt No '{}' not found. Cannot Update.".format(receiptNo)}

        return {'Is Error': False, 'Error Message': ""}

    def delete(self, receiptNo):
        if receiptNo in self.dict:
            del self.dict[receiptNo]
        else:
            return {'Is Error': True, 'Error Message': "Receipt No '{}' not found. Cannot Delete".format(receiptNo)}
        return {'Is Error': False, 'Error Message': ""}

    def dump(self):
        # Will dump all products data by returning 1 dictionary as output.
        return (self.dict)

    def update_receipt_line(self, receiptNo, invoiceNo, amountPaid):
        if receiptNo in self.dict:
            receiptLineItemList = []
            bUpdated = False
            for lineItem in self.dict[receiptNo]["Receipt Line Item"]:
                invoiceLineItem = {}
                if lineItem["Invoice No"] == invoiceNo:
                    invoiceLineItem["Invoice No"] = invoiceNo
                    invoiceLineItem["Amount Paid Here"] = amountPaid

                    receiptLineItemList.append(invoiceLineItem)
                    bUpdated = True
                else:
                    receiptLineItemList.append(lineItem)
            #print(receiptLineItemList)

            if bUpdated:
                receiptLineItemList,total = self.__updateLineItem(receiptLineItemList)
                self.dict[receiptNo]["Receipt Line Item"] = receiptLineItemList
                self.dict[receiptNo]["Total Received"] = total
            else:
                return {'Is Error': True, 'Error Message': "Receipt Code '{}' not found in Invoice No '{}'. Cannot Update.".format(receiptNo,invoic
        else:
            return {'Is Error': True, 'Error Message': "Receipt No '{}' not found. Cannot Update.".format(receiptNo)}

        return {'Is Error': False, 'Error Message': ""}

    def delete_receipt_line(self, receiptNo, invoiceNo):
        # The line item of this invoice number is updated to delete this product code.
        # Note that all the related data in the invoice must be updated such as Total, VAT, and Amount Due.
        # Returns dictionary {'Is Error': ___, 'Error Message': _____}
        if receiptNo in self.dict:
            total = 0
            receiptLineItemList = []
            bDeleted = False
            for lineItem in self.dict[receiptNo]["Receipt Line Item"]:
                if lineItem["Invoice No"] == invoiceNo:
                    bDeleted = True
                else:
                    receiptLineItemList.append(lineItem)

            if bDeleted:
                receiptLineItemList, total= self.__updateLineItem(receiptLineItemList)
                self.dict[receiptNo]["Receipt Line Item"] = receiptLineItemList
                self.dict[receiptNo]["Total Received"] = total

            else:
                return {'Is Error': True, 'Error Message': "Receipt Code '{}' not found in Invoice No '{}'. Cannot Delete.".format(receiptNo, invoi
        else:
            return {'Is Error': True, 'Error Message': "Receipt No '{}' not found. Cannot Delete.".format(receiptNo)}

        return {'Is Error': False, 'Error Message': ""}
```

หน้า API.py

API.py

```python
from helper_functions import *
#This file will contain all API functions calls exposed to outside world for users to use

 # function about payment method
 def create_payment_method(paymentMethod, code, name):
    result = paymentMethod.create(code, name)#returns error dictionary
    if result['Is Error']: #if error
        print(result['Error Message'])
    else:
        print('Payment Method Success.')
    return result #send result for caller program to use

def read_payment_method(paymentMethod, code):
    result = paymentMethod.read(code) #returns tuple of (error dict, data dict)
    if result[0]['Is Error']: #in case error
        print(result[0]['Error Message'])
    else:
        print(result[1])
    return result #send result for caller program to use

def update_payment_method(paymentMethod, code, newName):
    result = paymentMethod.update(code, newName) #returns error dictionary
    if result['Is Error']: #if error
        print(result['Error Message'])
    else:
        print('Payment Method  Success.')
    return result #send result for caller program to use

def delete_payment_method(paymentMethod, code):
    result = paymentMethod.delete(code)#returns error dictionary
    if result['Is Error']: #if error
        print(result['Error Message'])
    else:
        print('Product Delete Success.')
    return result #send result for caller program to use

def report_list_payment_method(paymentMethod):
    result = paymentMethod.dump()
    #printDictInCSVFormat(result, ('Code',), ('Name', 'Units'))
    print (result)
    return result #send result for caller program to use
# function about Product
def create_product(products, code, name, units):
    result = products.create(code, name, units)#returns error dictionary
    if result['Is Error']: #if error
        print(result['Error Message'])
    else:
        print('Product Create Success.')
    return result #send result for caller program to use

def read_product(products, code):
    result = products.read(code) #returns tuple of (error dict, data dict)
    if result[0]['Is Error']: #in case error
        print(result[0]['Error Message'])
    else:
        print(result[1])
    return result #send result for caller program to use

def update_product(products, code, newName, newUnits):
    result = products.update(code, newName, newUnits) #returns error dictionary
    if result['Is Error']: #if error
        print(result['Error Message'])
    else:
        print('Product Update Success.')
    return result #send result for caller program to use

def delete_product(products, code):
    result = products.delete(code)#returns error dictionary
    if result['Is Error']: #if error
        print(result['Error Message'])
    else:
        print('Product Delete Success.')
    return result #send result for caller program to use

def report_list_products(products):
    result = products.dump()
    #printDictInCSVFormat(result, ('Code',), ('Name', 'Units'))
    print (result)
    return result #send result for caller program to use

# function about Customer
def create_customer(customers, customerCode, customerName, address, creditLimit, country):
    result = customers.create(customerCode, customerName, address, creditLimit, country)#returns error dictionary
    if result['Is Error']: #if error
        print(result['Error Message'])
    else:
        print('Customer Create Success.')
    return result #send result for caller program to use

def read_customer(customers, customerCode):
    result = customers.read(customerCode) #returns tuple of (error dict, data dict)
    if result[0]['Is Error']: #in case error
        print(result[0]['Error Message'])
    else:
        print(result[1])
    return result #send result for caller program to use

def update_customer(customers, customerCode, newCustomerName, newAddress, newCreditLimit, newCountry):
    result = customers.update(customerCode, newCustomerName, newAddress, newCreditLimit, newCountry) #returns error dictionary
    if result['Is Error']: #if error
        print(result['Error Message'])
    else:
        print('Customer Update Success.')
    return result #send result for caller program to use

def delete_customer(customers, customerCode):
    result = customers.delete(customerCode)#returns error dictionary
    if result['Is Error']: #if error
        print(result['Error Message'])
    else:
        print('Customer Delete Success.')
    return result #send result for caller program to use

def report_list_all_customers(customers):
    result = customers.dump()
    printDictInCSVFormat(result, ('Customer Code',), ('Name', 'Address','Credit Limit', 'Country'))
    return result #send result for caller program to use

# function about Invoice
def create_invoice(invoices, invoiceNo, invoiceDate, customerCode, dueDate, invoiceLineTuplesList):
    result = invoices.create(invoiceNo, invoiceDate, customerCode, dueDate, invoiceLineTuplesList)#returns error dictionary
    if result['Is Error']: #if error
        print(result['Error Message'])
    else:
        print('Invoice Create Success.')
    return result #send result for caller program to use

def read_invoice(invoices, invoiceNo):
    result = invoices.read(invoiceNo) #returns tuple of (error dict, data dict)
    if result[0]['Is Error']: #in case error
        print(result[0]['Error Message'])
    else:
        print(result[1])
    return result #send result for caller program to use

def update_invoice(invoices, invoiceNo, newInvoiceDate, newCustomerCode, newDueDate, newInvoiceLineTuplesList):
    result = invoices.update(invoiceNo, newInvoiceDate, newCustomerCode, newDueDate, newInvoiceLineTuplesList) #returns error dictionary
    if result['Is Error']: #if error
        print(result['Error Message'])
    else:
        print('Invoice Update Success.')
    return result #send result for caller program to use

def delete_invoice(invoices, invoiceNo):
    result = invoices.delete(invoiceNo)#returns error dictionary
    if result['Is Error']: #if error
        print(result['Error Message'])
    else:
        print('Invoice Delete Success.')
    return result #send result for caller program to use

```

payment_medthod

Product

Customer

Invoice

ต่อ(งง)

```python
def update_invoice_line(invoices, invoiceNo, productCode, newQuantity, newUnitPrice):
    result = invoices.update_invoice_line(invoiceNo, productCode, newQuantity, newUnitPrice) #returns error dictionary
    if result['Is Error']: #if error
        print(result['Error Message'])
    else:
        print('Invoice Line Item Update Success.')
    return result #send result for caller program to use

def delete_invoice_line(invoices, invoiceNo, productCode):
    result = invoices.delete_invoice_line(invoiceNo, productCode) #returns error dictionary
    if result['Is Error']: #if error
        print(result['Error Message'])
    else:
        print('Invoice Line Item Delete Success.')
    return result #send result for caller program to use

def report_list_all_invoices(invoices, customers, products):
    # Will dump all invoices data and return 1 dictionary as a result (with header and line item joined).
    # Please show the customer name and product name also.
    # A helper function such as def print_tabular_dictionary(tabularDictionary) can then be called to print this in a tabular (table-like) form with column headings and data.

    allInvoice = invoices.dump()
    # re-format result by add Customer Name in object and Product Name in line item
    result = {}
    for invoiceNo, invoiceDetail in allInvoice.items():
        newValue = {}
        for invoiceColume, invoiceData in invoiceDetail.items():
            if invoiceColume == "Customer Code":
                newValue[invoiceColume] = invoiceData
                customer = customers.read(invoiceDetail["Customer Code"])
                if customer[0]['Is Error']:
                    newValue["Customer Name"] = ""
                else:
                    newValue["Customer Name"] = customer[1]["Name"]
            elif invoiceColume == "Items List":
                newLineItemList = []
                for lineItem in invoiceDetail['Items List']:
                    newLineItem = {}
                    for lineItemColume, lineItemData in lineItem.items():
                        if lineItemColume == "Product Code":
                            newLineItem[lineItemColume] = lineItemData

                            product = products.read(lineItem["Product Code"])
                            if product[0]['Is Error']:
                                newLineItem["Product Name"] = ""
                            else:
                                newLineItem["Product Name"] = product[1]["Name"]
                        else:
                            newLineItem[lineItemColume] = lineItemData
                    newLineItemList.append (newLineItem)
                newValue[invoiceColume] = newLineItemList
            else:
                newValue[invoiceColume] = invoiceData
        result[invoiceNo] = newValue
    #print (result)
    printDictInCSVFormat(result, ('Invoice No',), ('Invoice Date', 'Customer Code','Due Date','Total','VAT','Amount Due','Items List'))
    return result #send result for caller program to use

def report_products_sold(invoices, products, dateStart, dateEnd):
    # Will return 2 dictionaries:
    # 1) a dictionary as list of in products sold in the given date range in tabular format of: Product Code, Product Name, Total Quantity Sold, Total Value Sold. Here, (product code) a
    # And 2) a second dictionary of the footer will also be returned containing: t the end also show the sum of Total Value Sold.
    allInvoice = invoices.dump()
    result = {}
    result2 = {}
    sumTotalValueSold = 0
    for key, value in allInvoice.items():
        if (dateStringInDateRange(value['Date'], dateStart, dateEnd)):
            for lineItem in value['Items List']:
                primaryKey = lineItem['Product Code']
                if primaryKey in result:
                    result[primaryKey]['Total Quantity Sold'] += lineItem['Quantity']
                    result[primaryKey]['Total Value Sold'] += lineItem['Extended Price']
                else:
                    product = products.read(lineItem["Product Code"])
                    if product[0]['Is Error']:
                        productName = ""
                    else:
                        productName = product[1]["Name"]
                    result[primaryKey] = {'Product Code':lineItem['Product Code'], 'Product Name':productName,'Total Quantity Sold':lineItem['Quantity'],'Total Value Sold':lineItem['Ext
                sumTotalValueSold += lineItem['Extended Price']

    result2['Sum of'] = {'Total Value Sold':sumTotalValueSold}
    printDictInCSVFormat(result, (None), ('Product Code','Product Name', 'Total Quantity Sold', 'Total Value Sold'))
    printDictInCSVFormat(result2, (None), ('Total Value Sold',))
    return result.values(), result2

def report_customer_products_sold_list(invoices, products, customers, dateStart, dateEnd):
    # Will return 2 dictionaries:
    # 1) a dictionary as list of customers and list the products sold to them in the given date range in this format:  Customer Code, Customer Name, Product Code,  Product Name, Invoic
    # And 2) a second footer dictionary showing:  At the end also show the sum of Quantity Sold and sum of Value Sold.
    allInvoice = invoices.dump()
    result = {}
    result2 = {}
    sumTotalQuantitySold = 0
    sumTotalValueSold = 0
    for key, value in allInvoice.items():
        invoiceNo = key
        if (dateStringInDateRange(value['Date'], dateStart, dateEnd)):
            for lineItem in value['Items List']:
                primaryKey = value['Customer Code'] + lineItem['Product Code'] + invoiceNo
                if primaryKey in result:
                    result[primaryKey]['Quantity Sold'] += lineItem['Quantity']
                    result[primaryKey]['Value Sold'] += lineItem['Extended Price']
                else:
                    product = products.read(lineItem["Product Code"])
                    if product[0]['Is Error']:
                        productName = ""
                    else:
                        productName = product[1]["Name"]

                    customer = customers.read(value['Customer Code'])
                    if customer[0]['Is Error']:
                        customerName = ""
                    else:
                        customerName = customer[1]["Name"]
                    result[primaryKey] = {'Customer Code':value['Customer Code'],'Customer Name':customerName,'Product Code':lineItem['Product Code'], 'Product Name':productName,'Invoi
                sumTotalQuantitySold += lineItem['Quantity']
                sumTotalValueSold += lineItem['Extended Price']

    result2['Sum of'] = {'Quantity Sold':sumTotalQuantitySold,'Value Sold':sumTotalValueSold}
    printDictInCSVFormat(result, (None), ('Customer Code','Customer Name', 'Product Code', 'Product Name', 'Invoice No', 'Invoice Date', 'Quantity Sold', 'Value Sold'))
    printDictInCSVFormat(result2, (None), ('Quantity Sold','Value Sold'))
    return result.values(), result2

def report_customer_products_sold_total(invoices, products, customers, dateStart, dateEnd):
    allInvoice = invoices.dump()
    result = {}
    result2 = {}
    sumTotalQuantitySold = 0
    sumTotalValueSold = 0
    for key, value in allInvoice.items():
        invoiceNo = key
        if (dateStringInDateRange(value['Date'], dateStart, dateEnd)):
            for lineItem in value['Items List']:
                primaryKey = value['Customer Code'] + lineItem['Product Code']
                if primaryKey in result:
                    result[primaryKey]['Total Quantity Sold'] += lineItem['Quantity']
                    result[primaryKey]['Total Value Sold'] += lineItem['Extended Price']
                else:
                    product = products.read(lineItem["Product Code"])
                    if product[0]['Is Error']:
                        productName = ""
                    else:
                        productName = product[1]["Name"]

                    customer = customers.read(value['Customer Code'])
                    if customer[0]['Is Error']:
                        customerName = ""
                    else:
                        customerName = customer[1]["Name"]
                    result[primaryKey] = {'Customer Code':value['Customer Code'],'Customer Name':customerName,'Product Code':lineItem['Product Code'], 'Product Name':productName,'Total
                sumTotalQuantitySold += lineItem['Quantity']
                sumTotalValueSold += lineItem['Extended Price']

    result2['Sum of'] = {'Total Quantity Sold':sumTotalQuantitySold,'Total Value Sold':sumTotalValueSold}
    printDictInCSVFormat(result, (None), ('Customer Code','Customer Name', 'Product Code', 'Product Name', 'Total Quantity Sold', 'Total Value Sold'))
    printDictInCSVFormat(result2, (None), ('Total Quantity Sold','Total Value Sold'))
    return result.values(), result2

```

Invoice

```python
def report_unpaid_invoices(invoices,customers,receipts):
    total = 0
    allreceipt = receipts.dump()
    result = {}
    for receiptKey, receiptCol in allreceipt.items():
        for receiptLine in receiptCol['Receipt Line Item']:
            if receiptLine['Invoice No'] not in result: # ถ้ายังไม่เคยเจอ Invoice นี้เลยสร้างมาใหม่
                newValue = {}
                newValue['Invoice Amount Reveived'] = receiptLine['Amount Paid Here']
                invoice = invoices.read(receiptLine['Invoice No'])
                if invoice[0]['Is Error']:
                    newValue["Invoice Date"] = ""
                    newValue["Customer Name"] = ""
                    newValue["Invoice Amount Due"] = ""
                    newValue["Invoice Amount Not Paid"] = ""
                else:
                    newValue["Invoice Date"] = invoice[1]['Invoice Date']
                    newValue["Invoice Amount Due"] = invoice[1]['Amount Due']
                    newValue["Invoice Amount Not Paid"] = newValue["Invoice Amount Due"] - newValue['Invoice Amount Reveived']
                    customer = customers.read(invoice[1]['Customer Code'])
                    if customer[0]['Is Error']:
                        newValue["Customer Name"] = ""
                    else:
                        newValue["Customer Name"] = customer[1]['Name']
                result[receiptLine['Invoice No']] = newValue
            else:
                result[receiptLine['Invoice No']]['Invoice Amount Reveived'] += receiptLine['Amount Paid Here']
                result[receiptLine['Invoice No']]["Invoice Amount Not Paid"] = result[receiptLine['Invoice No']]["Invoice Amount Due"] - result[receiptLine['Invoice No']]['Invoice Amoun
    printDictInCSVFormat(result, ('Invoice No',), ('Invoice Date', 'Customer Name','Invoice Amount Due','Invoice Amount Reveived','Invoice Amount Not Paid'))

    for line in result:
        total += result[line]["Invoice Amount Not Paid"]

    print("Total Debt: ",total)



# function about Receipt
def create_receipt(receipt, receiptNo, receiptDate, customerCode, paymentMethod, paymentReference, remark, receiptLineItemList):
    result = receipt.create(receiptNo, receiptDate, customerCode, paymentMethod, paymentReference, remark, receiptLineItemList)#returns error dictionary
    if result['Is Error']: #if error
        print(result['Error Message'])
    else:
        print('Receipt Create Success.')
    return result #send result for caller program to use

def read_receipt(receipt, receiptNo):
    result = receipt.read(receiptNo) #returns tuple of (error dict, data dict)
    if result[0]['Is Error']: #in case error
        print(result[0]['Error Message'])
    else:
        print(result[1])
    return result #send result for caller program to use

def update_receipt(receipt, receiptNo, newReceiptDate, newCustomerCode, newPaymentMethod, newPaymentReference, newRemark, newReceiptLineItemList):
    result = receipt.update(receiptNo, newReceiptDate, newCustomerCode, newPaymentMethod, newPaymentReference, newRemark, newReceiptLineItemList) #returns error dictionary
    if result['Is Error']: #if error
        print(result['Error Message'])
    else:
        print('Receipt Update Success.')
    return result #send result for caller program to use

def delete_receipt(receipt, receiptNo):
    result = receipt.delete(receiptNo)#returns error dictionary
    if result['Is Error']: #if error
        print(result['Error Message'])
    else:
        print('Reciept Delete Success.')
    return result #send result for caller program to use

def update_receipt_line(Receipt, receiptNo, invoiceNo, amountPaid):
    result = Receipt.update_receipt_line(receiptNo, invoiceNo, amountPaid) #returns error dictionary
    if result['Is Error']: #if error
        print(result['Error Message'])
    else:
        print('Invoice Line Item Update Success.')
    return result #send result for caller program to use

def delete_receipt_line(receipt, receiptNo, invoiceNo):
    result = receipt.delete_receipt_line(receiptNo, invoiceNo) #returns error dictionary
    if result['Is Error']: #if error
        print(result['Error Message'])
    else:
        print('Receipt Line Item Delete Success.')
    return result #send result for caller program to use

def report_list_all_receipt(receipt, invoices, customers):
    allreceipt = receipt.dump()
    # re-format result by add Customer Name in object and Product Name in line item
    result = {}
    for receiptNo, receiptDetail in allreceipt.items():
        newValue = {}
        for receiptColume, receiptData in receiptDetail.items():
            if receiptColume == "Customer Code":
                newValue[receiptColume] = receiptData
                customer = customers.read(receiptDetail["Customer Code"])
                if customer[0]['Is Error']:
                    newValue["Customer Name"] = ""
                else:
                    newValue["Customer Name"] = customer[1]["Name"]

            elif receiptColume == "Receipt Line Item":
                newLineItemList = []
                for lineItem in receiptDetail['Receipt Line Item']:
                    newLineItem = {}
                    for lineItemColume, lineItemData in lineItem.items():
                        if lineItemColume == "Invoice No":
                            newLineItem[lineItemColume] = lineItemData
                            invoice = invoices.read(lineItem["Invoice No"]) #join

                            if invoice[0]['Is Error']:
                                newLineItem[1]["Invoice Date"] = ""

                            else:
                                newLineItem["Invoice Date"] = invoice[1]["Invoice Date"]
                        else:
                            newLineItem[lineItemColume] = lineItemData

                    newLineItemList.append (newLineItem)
                newValue[receiptColume] = newLineItemList
            else:
                newValue[receiptColume] = receiptData
        result[receiptNo] = newValue
    printDictInCSVFormat(result, ('Receipt No',), ('Date', 'Customer Code','Customer Name','Payment Method','Payment Reference','Remarks','Total Received','Receipt Line Item'))
    return result #send result for caller program to use
```

unpaid

receipt

หน้า lab2_mainfunction.py

lab2_mainfunction.py

```
from helper_functions import *
from Invoice import *
from Receipt import *
from Product import *
from Customer import *
from PaymentMethod import *
from API import *


#Replace these lines with main function
def main():
#main function begins here
    try:

        products = Product() # create object products from class Product. Starts as blank dict.
        #'HD01': {'Name': 'Seagate HDD 80 GB', 'Units': 'PCS'},
        # 'HD02': {'Name': 'IBM HDD 60 GB', 'Units': 'PCS'},
        # 'INT01': {'Name': 'Intel Pentium IV 3.6 GHz', 'Units': 'PCS'}}

        create_product(products, "HD01", "Seagate HDD 80 GB", "PCS")
        create_product(products, "HD02", "IBM HDD 60 GB", "PCS")
        create_product(products, "INT01", "Intle Pentium IV 3.6 GHz", "PCS")
        create_product(products, "INT99", "Intle Pentium V 4.2 GHz", "PCS")
        report_list_products(products)
        waitKeyPress("Above are results for creating 4 products.")

        read_product(products, "HD01")
        read_product(products, "HD99") #error
        #correct the spelling error of "Intle":
        update_product(products, newName = "Intel Pentium IV 3.6 GHz", newUnits = "PCS",
                        code = "INT01")
        update_product(products, "INT99", "Intel Pentium V 4.2 GHz", "PCS")
        report_list_products (products)
        waitKeyPress("Results after 2 reads, 2 updates to correct Intle spelling.")

        delete_product(products, "INT33") #error
        delete_product(products, "INT99")
        report_list_products(products)
        waitKeyPress("Results after deleting INT33 (not exist error) and INT99.")

        # pretty print a dictionary (in helper functions):
        printDictData(products.dict)
        waitKeyPress("Above is dictionary printed in better format.")

        # pretty print in column format a dictionary (in helper functions):
        printDictInCSVFormat(products.dict, ('Code',), ('Name', 'Units'))
        waitKeyPress("Above is dictionary printed in csv format for copy/paste to excel.")

        """
        #shows in case of untrapped exception:
        result = products.dict["HDD5"]
        waitKeyPress("There will be error and exit before you see this.")
        """

        customers = Customer()
        create_customer(customers, "Sam", "Sam Co., Ltd.", "122 Bangkok", 500000, "Thailand")
        create_customer(customers, "CP", "Charoen Pokaphan", "14 Sukhumvit, Bangkok", 2000000, "Thailand")
        report_list_all_customers(customers)
        waitKeyPress("Above are results for creating 2 customers.")
        print(" ")

        read_customer(customers, "IT City")#error
        read_customer(customers, "Sam")
        update_customer(customers, newCustomerName = "CPALL", newAddress = "123 Bangkok", newCreditLimit = 100000, newCountry = "Thailand", customerCode = "CP")
        delete_customer(customers, "CP1")#not found
        #delete_customer(customers, "CP")
        report_list_all_customers(customers)
        waitKeyPress("Results after 2 reads, and update and delete customer CP")
        print(" ")

        invoices = Invoice()
        create_invoice(invoices, invoiceNo="INT100/20", invoiceDate="2020-01-02", customerCode="Sam", dueDate=None, invoiceLineTuplesList=[{"Product Code":"HD01","Quantity":2,"Unit Price":3000},{"Product Code":"HD02","Quantity":1,
        create_invoice(invoices, "INT101/20", "2020-01-04", "CP", None, [{"Product Code":"HD02","Quantity":1,"Unit Price":2000}])
        report_list_all_invoices(invoices, customers, products)
        waitKeyPress("Above are results for creating 2 invoices and line item.")
        print(" ")


        read_invoice(invoices, "INT100/20")
        update_invoice(invoices, invoiceNo="INT100/20", newInvoiceDate="2020-01-03", newCustomerCode="Sam", newDueDate=None, newInvoiceLineTuplesList=[{"Product Code":"HD01","Quantity":2,"Unit Price":3000},{"Product Code":"HD02","
        #delete_invoice(invoices, "INT101/19")
        report_list_all_invoices(invoices, customers, products)
        waitKeyPress("Results after read, update and delete Invoice")
        print(" ")


        update_invoice_line(invoices, "INT100/20", "HD02", 8, 1000)
        report_list_all_invoices(invoices, customers, products)
        waitKeyPress("Results after update Invoice Line Item")
        print(" ")

        #delete_invoice_line(invoices, "INT101/20", "HD02")
        report_list_all_invoices(invoices, customers, products)
        waitKeyPress("Results after delete Invoice Line Item")
        print(" ")

        #Test receipt functions
        receipts = Receipt()
        #Create receipt
        print("*"*50,"Create Receipt","*"*50)
        print(" ")
        create_receipt(receipts,'RCT1001/20','2020-02-04','CP','Cash','Nothing','Paid all invoices partially',[{'Invoice No':'INT100/20','Amount Paid Here':100},{'Invoice No':'INT101/20','Amount Paid Here':200}])
        create_receipt(receipts,'RCT1002/20','2020-02-05','Sam','Credit Card','Master Card, Citibank','Partially paid on INT101/20',[{'Invoice No':'INT100/20','Amount Paid Here':8560},{'Invoice No':'INT101/20','Amount Paid Here':1
        create_receipt(receipts,'RCT1003/20','2020-02-06','CP','Debit Card','Debit Card','This will later be deleted',[{'Invoice No':'INT100/20','Amount Paid Here':10},{'Invoice No':'INT101/20','Amount Paid Here':20}])
        report_list_all_receipt(receipts,invoices,customers)
        waitKeyPress("Results of creating 3 receipts: RCT1001/20, RCT1002/20, and RCT1003/20")

        #Read receipt
        read_receipt(receipts,'RCT1001/20')
        read_receipt(receipts,'RCT1003/20')
        read_receipt(receipts,'RCT1005/20')
        report_list_all_receipt(receipts,invoices,customers)
        waitKeyPress("Results of reading 3 receipts: RCT1001/20 (successfully), RCT1003/20 (successfully), and RCT1005/20 (unsuccessfully)")
        print(" ")

        #Update receipt
        print("*"*50,"Update Receipt","*"*50)
        print(" ")
        update_receipt(receipts,'RCT1002/20','2020-02-06','Sam','Debit Card','VISA card','Partially paid on INT100/20 and INT101/20',[{'Invoice No':"INT100/20",'Amount Paid Here':8000},{'Invoice No':"INT101/20",'Amount Paid Here':
        update_receipt(receipts,'RCT1004/20','1999-12-31','Sam','Credit Card','Master Card, Citibank','Partially paid on INT101/20',[{'Invoice No':"INT100/20",'Amount Paid Here':8560},{'Invoice No':"INT101/20",'Amount Paid Here':1
        report_list_all_receipt(receipts,invoices,customers)
        waitKeyPress("Results of updating 2 receipts: RCT1002/20 (successfully) and RCT1004/20 (unsuccessfully)")
        print(" ")

        #Delete receipt
        print("*"*50,"delete_receipt ","*"*50)
        print(" ")
        delete_receipt(receipts,'RCT1003/20')
        delete_receipt(receipts,'RCT1005/20')
        report_list_all_receipt(receipts,invoices,customers)
        waitKeyPress("Results of deleting 2 receipts: RCT1003/20 (successfully) and RCT1005/20 (unsuccessfully)")
        print(" ")

        #Report unpaid invoice 1
        report_unpaid_invoices(invoices,customers,receipts)
        waitKeyPress("Report unpaid invoice 1")
        print(" ")

        #Update receipt line
        print("*"*50,"Update receipt line","*"*50)
        print(" ")
        update_receipt_line(receipts,'RCT1001/20','INT100/20',200)
        update_receipt_line(receipts,'RCT1005/20','INT100/20',2000)
        update_receipt_line(receipts,'RCT1001/20','INT103/20',5000)
        report_list_all_receipt(receipts,invoices,customers)
        waitKeyPress("Result of updating 3 receipt line, first successfully, others unsuccessfully")
        print(" ")

        #Delete receipt line
        print("*"*50,"Delete receipt line","*"*50)
        print(" ")
        delete_receipt_line(receipts,'RCT1001/20','INT101/20')
        delete_receipt_line(receipts,'RCT1003/20','INT100/20')
        delete_receipt_line(receipts,'RCT1001/20','INT103/20')
        report_list_all_receipt(receipts,invoices,customers)
        waitKeyPress("Result of deleting 3 receipt line, first successfully, others unsuccessfully")
        print(" ")

        #print ("\nTest Report")
        #report_products_sold(invoices, products, '2020-01-01', '2020-01-31')
        #report_customer_products_sold_list(invoices, products, customers, '2020-01-01', '2020-01-31')
        #report_customer_products_sold_total(invoices, products, customers, '2020-01-01', '2020-01-31')
        report_unpaid_invoices(invoices,customers,receipts)
        print(" ")


    except: #this traps for unexpected system errors
        print ("Unexpected error:", sys.exc_info()[0])
        raise # this line can be erased. It is here to raise another error so you can see which line to debug.
    else:
        print("Normal Termination.   Goodbye!")
#main function ends




if __name__ == "__main__":
    main()
```

เพิ่มเข้ามา

C R U D — product

C R U D — Customer

C R U D — invoice

receipt

ได้เปลี่ยนจาก report_list_all_receipts เป็น report_list_all_receipt (โดยตัด s หลัง receipt ออกไป)

เพิ่มส่วนนี้

- มีการเพิ่ม หัวข้อของพวก CRUD ด้วย การปริ้น ตัวอย่าง *** create receipt ***

"แสดงผล"

```
Product Create Success.
Product Create Success.
Product Create Success.
Product Create Success.
{'HD01': {'Name': 'Seagate HDD 80 GB', 'Units': 'PCS'}, 'HD02': {'Name': 'IBM HDD 60 GB', 'Units': 'PCS'}, 'INT01': {'Name': 'Intle Pentium IV 3.6 GHz', 'Units': 'PCS'}, 'INT99': {'Name': 'Intle Pentium V 4.2 GHz', 'Units': 'PCS'}}
Above are results for creating 4 products. (Press a key to continue).
{'Name': 'Seagate HDD 80 GB', 'Units': 'PCS'}
Product Code 'HD99' not found. Cannot Read.
Product Update Success.
Product Update Success.
{'HD01': {'Name': 'Seagate HDD 80 GB', 'Units': 'PCS'}, 'HD02': {'Name': 'IBM HDD 60 GB', 'Units': 'PCS'}, 'INT01': {'Name': 'Intel Pentium IV 3.6 GHz', 'Units': 'PCS'}, 'INT99': {'Name': 'Intel Pentium V 4.2 GHz', 'Units': 'PCS'}}
Results after 2 reads, 2 updates to correct Intle speiling. (Press a key to continue).
Product Code 'INT33' not found. Cannot Delete
Product Delete Success.
{'HD01': {'Name': 'Seagate HDD 80 GB', 'Units': 'PCS'}, 'HD02': {'Name': 'IBM HDD 60 GB', 'Units': 'PCS'}, 'INT01': {'Name': 'Intel Pentium IV 3.6 GHz', 'Units': 'PCS'}}
Results after deleting INT33 (not exist error) and INT99. (Press a key to continue).

HD01 : {'Name': 'Seagate HDD 80 GB', 'Units': 'PCS'}
HD02 : {'Name': 'IBM HDD 60 GB', 'Units': 'PCS'}
INT01 : {'Name': 'Intel Pentium IV 3.6 GHz', 'Units': 'PCS'}
Above is dictionary printed in better format. (Press a key to continue).

Code, Name, Units
HD01, Seagate HDD 80 GB, PCS
HD02, IBM HDD 60 GB, PCS
INT01, Intel Pentium IV 3.6 GHz, PCS
Above is dictionary printed in csv format for copy/paste to excel. (Press a key to continue).
Customer Create Success.
Customer Create Success.

Customer Code, Name, Address, Credit Limit, Country
Sam, Sam Co., Ltd., 122 Bangkok, 500000, Thailand
CP, Charoen Pokaphan, 14 Sukhumvit, Bangkok, 2000000, Thailand
Above are results for creating 2 customers. (Press a key to continue).

Customer Code 'IT City' not found. Cannot Read.
{'Name': 'Sam Co., Ltd.', 'Address': '122 Bangkok', 'Credit Limit': 500000, 'Country': 'Thailand'}
Customer Update Success.
Customer Code 'CP1' not found. Cannot Delete

Customer Code, Name, Address, Credit Limit, Country
Sam, Sam Co., Ltd., 122 Bangkok, 500000, Thailand
CP, CPALL, 123 Bangkok, 100000, Thailand
Results after 2 reads, and update and delete customer CP (Press a key to continue).
```

```
Invoice Create Success.
Invoice Create Success.

Invoice No, Invoice Date, Customer Code, Due Date, Total, VAT, Amount Due, Items List
INT100/20, 2020-01-02, Sam, None, 8000, 560.0, 8560.0, [{'Product Code': 'HD01', 'Product Name': 'Seagate HDD 80 GB', 'Quantity': 2, 'Unit Price': 3000, 'Extended Price': 6000}, {'Product Code': 'HD02', 'Product Name': 'IBM HDD 60 GB', 'Quantity': 1, 'Unit Price': 2000, 'Extended Price': 2000}]
INT101/20, 2020-01-04, CP, None, 2000, 140.0, 2140.0, [{'Product Code': 'HD02', 'Product Name': 'IBM HDD 60 GB', 'Quantity': 1, 'Unit Price': 2000, 'Extended Price': 2000}]
Above are results for creating 2 invoices and line item. (Press a key to continue).

{'Invoice Date': '2020-01-02', 'Customer Code': 'Sam', 'Due Date': None, 'Total': 8000, 'VAT': 560.0, 'Amount Due': 8560.0, 'Items List': [{'Product Code': 'HD01', 'Quantity': 2, 'Unit Price': 3000, 'Extended Price': 6000}, {'Product Code': 'HD02', 'Quantity': 1, 'Unit Price': 2000, 'Extended Price': 2000}]}
Invoice Update Success.

Invoice No, Invoice Date, Customer Code, Due Date, Total, VAT, Amount Due, Items List
INT100/20, 2020-01-03, Sam, None, 8000, 560.0, 8560.0, [{'Product Code': 'HD01', 'Product Name': 'Seagate HDD 80 GB', 'Quantity': 2, 'Unit Price': 3000, 'Extended Price': 6000}, {'Product Code': 'HD02', 'Product Name': 'IBM HDD 60 GB', 'Quantity': 1, 'Unit Price': 2000, 'Extended Price': 2000}]
INT101/20, 2020-01-04, CP, None, 2000, 140.0, 2140.0, [{'Product Code': 'HD02', 'Product Name': 'IBM HDD 60 GB', 'Quantity': 1, 'Unit Price': 2000, 'Extended Price': 2000}]
Results after read, update and delete Invoice (Press a key to continue).

Invoice Line Item Update Success.

Invoice No, Invoice Date, Customer Code, Due Date, Total, VAT, Amount Due, Items List
INT100/20, 2020-01-03, Sam, None, 14000, 980.0, 14980.0, [{'Product Code': 'HD01', 'Product Name': 'Seagate HDD 80 GB', 'Quantity': 2, 'Unit Price': 3000, 'Extended Price': 6000}, {'Product Code': 'HD02', 'Product Name': 'IBM HDD 60 GB', 'Quantity': 8, 'Unit Price': 1000, 'Extended Price': 8000}]
INT101/20, 2020-01-04, CP, None, 2000, 140.0, 2140.0, [{'Product Code': 'HD02', 'Product Name': 'IBM HDD 60 GB', 'Quantity': 1, 'Unit Price': 2000, 'Extended Price': 2000}]
Results after update Invoice Line Item (Press a key to continue).


Invoice No, Invoice Date, Customer Code, Due Date, Total, VAT, Amount Due, Items List
INT100/20, 2020-01-03, Sam, None, 14000, 980.0, 14980.0, [{'Product Code': 'HD01', 'Product Name': 'Seagate HDD 80 GB', 'Quantity': 2, 'Unit Price': 3000, 'Extended Price': 6000}, {'Product Code': 'HD02', 'Product Name': 'IBM HDD 60 GB', 'Quantity': 8, 'Unit Price': 1000, 'Extended Price': 8000}]
INT101/20, 2020-01-04, CP, None, 2000, 140.0, 2140.0, [{'Product Code': 'HD02', 'Product Name': 'IBM HDD 60 GB', 'Quantity': 1, 'Unit Price': 2000, 'Extended Price': 2000}]
Results after delete Invoice Line Item (Press a key to continue).
```

```
************************************************** Create Receipt **************************************************

Receipt Create Success.
Receipt Create Success.
Receipt Create Success.

Receipt No, Date, Customer Code, Customer Name, Payment Method, Payment Reference, Remarks, Total Received, Receipt Line Item
RCT1001/20, 2020-02-04, CP, CPALL, Cash, Nothing, Paid all invoices partially, 300, [{'Invoice No': 'INT100/20', 'Invoice Date': '2020-01-03', 'Amount Paid Here': 100}, {'Invoice No': 'INT101/20', 'Invoice Date': '2020-01-04', 'Amount Paid Here': 200}]
RCT1002/20, 2020-02-05, Sam, Sam Co., Ltd., Credit Card, Master Card, Citibank, Partially paid on INT101/20, 10000, [{'Invoice No': 'INT100/20', 'Invoice Date': '2020-01-03', 'Amount Paid Here': 8560}, {'Invoice No': 'INT101/20', 'Invoice Date': '2020-01-04', 'Amount Paid Here': 1440}]
RCT1003/20, 2020-02-06, CP, CPALL, Debit Card, Debit Card, This will later be deleted, 30, [{'Invoice No': 'INT100/20', 'Invoice Date': '2020-01-03', 'Amount Paid Here': 10}, {'Invoice No': 'INT101/20', 'Invoice Date': '2020-01-04', 'Amount Paid Here': 20}]
Results of creating 3 receipts: RCT1001/20, RCT1002/20, and RCT1003/20 (Press a key to continue).
{'Date': '2020-02-04', 'Customer Code': 'CP', 'Payment Method': 'Cash', 'Payment Reference': 'Nothing', 'Total Received': 300, 'Remarks': 'Paid all invoices partially', 'Receipt Line Item': [{'Invoice No': 'INT100/20', 'Amount Paid Here': 100}, {'Invoice No': 'INT101/20', 'Amount Paid Here': 200}]}
{'Date': '2020-02-06', 'Customer Code': 'CP', 'Payment Method': 'Debit Card', 'Payment Reference': 'Debit Card', 'Total Received': 30, 'Remarks': 'This will later be deleted', 'Receipt Line Item': [{'Invoice No': 'INT100/20', 'Amount Paid Here': 10}, {'Invoice No': 'INT101/20', 'Amount Paid Here': 20}]}
Receipt No 'RCT1005/20' not found. Cannot Read.

Receipt No, Date, Customer Code, Customer Name, Payment Method, Payment Reference, Remarks, Total Received, Receipt Line Item
RCT1001/20, 2020-02-04, CP, CPALL, Cash, Nothing, Paid all invoices partially, 300, [{'Invoice No': 'INT100/20', 'Invoice Date': '2020-01-03', 'Amount Paid Here': 100}, {'Invoice No': 'INT101/20', 'Invoice Date': '2020-01-04', 'Amount Paid Here': 200}]
RCT1002/20, 2020-02-05, Sam, Sam Co., Ltd., Credit Card, Master Card, Citibank, Partially paid on INT101/20, 10000, [{'Invoice No': 'INT100/20', 'Invoice Date': '2020-01-03', 'Amount Paid Here': 8560}, {'Invoice No': 'INT101/20', 'Invoice Date': '2020-01-04', 'Amount Paid Here': 1440}]
RCT1003/20, 2020-02-06, CP, CPALL, Debit Card, Debit Card, This will later be deleted, 30, [{'Invoice No': 'INT100/20', 'Invoice Date': '2020-01-03', 'Amount Paid Here': 10}, {'Invoice No': 'INT101/20', 'Invoice Date': '2020-01-04', 'Amount Paid Here': 20}]
Results of reading 3 receipts: RCT1001/20 (successfully), RCT1003/20 (successfully), and RCT1005/20 (unsuccessfully) (Press a key to continue).

************************************************** Update Receipt **************************************************

Receipt Update Success.
Receipt No 'RCT1004/20' not found. Cannot Update.

Receipt No, Date, Customer Code, Customer Name, Payment Method, Payment Reference, Remarks, Total Received, Receipt Line Item
RCT1001/20, 2020-02-04, CP, CPALL, Cash, Nothing, Paid all invoices partially, 300, [{'Invoice No': 'INT100/20', 'Invoice Date': '2020-01-03', 'Amount Paid Here': 100}, {'Invoice No': 'INT101/20', 'Invoice Date': '2020-01-04', 'Amount Paid Here': 200}]
RCT1002/20, 2020-02-06, Sam, Sam Co., Ltd., Debit Card, VISA card, Partially paid on INT100/20 and INT101/20, 9400, [{'Invoice No': 'INT100/20', 'Invoice Date': '2020-01-03', 'Amount Paid Here': 8000}, {'Invoice No': 'INT101/20', 'Invoice Date': '2020-01-04', 'Amount Paid Here': 1400}]
RCT1003/20, 2020-02-06, CP, CPALL, Debit Card, Debit Card, This will later be deleted, 30, [{'Invoice No': 'INT100/20', 'Invoice Date': '2020-01-03', 'Amount Paid Here': 10}, {'Invoice No': 'INT101/20', 'Invoice Date': '2020-01-04', 'Amount Paid Here': 20}]
Results of updating 2 receipts: RCT1002/20 (successfully) and RCT1004/20 (unsuccessfully) (Press a key to continue).
```

```
************************************************** delete_receipt  **************************************************

Reciept Delete Success.
Receipt No 'RCT1005/20' not found. Cannot Delete

Receipt No, Date, Customer Code, Customer Name, Payment Method, Payment Reference, Remarks, Total Received, Receipt Line Item
RCT1001/20, 2020-02-04, CP, CPALL, Cash, Nothing, Paid all invoices partially, 300, [{'Invoice No': 'INT100/20', 'Invoice Date': '2020-01-03', 'Amount Paid Here': 100}, {'Invoice No': 'INT101/20', 'Invoice Date': '2020-01-04', 'Amount Paid Here': 200}]
RCT1002/20, 2020-02-06, Sam, Sam Co., Ltd., Debit Card, VISA card, Partially paid on INT100/20 and INT101/20, 9400, [{'Invoice No': 'INT100/20', 'Invoice Date': '2020-01-03', 'Amount Paid Here': 8000}, {'Invoice No': 'INT101/20', 'Invoice Date': '2020-01-04', 'Amount Paid Here': 1400}]
Results of deleting 2 receipts: RCT1003/20 (successfully) and RCT1005/20 (unsuccessfully) (Press a key to continue).


Invoice No, Invoice Date, Customer Name, Invoice Amount Due, Invoice Amount Reveived, Invoice Amount Not Paid
INT100/20, 2020-01-03, Sam Co., Ltd., 14980.0, 8100, 6880.0
INT101/20, 2020-01-04, CPALL, 2140.0, 1600, 540.0
Total Debt:  7420.0
Report unpaid invoice 1 (Press a key to continue).

************************************************** Update receipt line **************************************************

Invoice Line Item Update Success.
Receipt No 'RCT1005/20' not found. Cannot Update.
Receipt Code 'RCT1001/20' not found in Invoice No 'INT103/20'. Cannot Update.

Receipt No, Date, Customer Code, Customer Name, Payment Method, Payment Reference, Remarks, Total Received, Receipt Line Item
RCT1001/20, 2020-02-04, CP, CPALL, Cash, Nothing, Paid all invoices partially, 400, [{'Invoice No': 'INT100/20', 'Invoice Date': '2020-01-03', 'Amount Paid Here': 200}, {'Invoice No': 'INT101/20', 'Invoice Date': '2020-01-04', 'Amount Paid Here': 200}]
RCT1002/20, 2020-02-06, Sam, Sam Co., Ltd., Debit Card, VISA card, Partially paid on INT100/20 and INT101/20, 9400, [{'Invoice No': 'INT100/20', 'Invoice Date': '2020-01-03', 'Amount Paid Here': 8000}, {'Invoice No': 'INT101/20', 'Invoice Date': '2020-01-04', 'Amount Paid Here': 1400}]
Result of updating 3 receipt line, first successfully, others unsuccessfully (Press a key to continue).

************************************************** Delete receipt line **************************************************

Receipt Line Item Delete Success.
Receipt No 'RCT1003/20' not found. Cannot Delete.
Receipt Code 'RCT1001/20' not found in Invoice No 'INT103/20'. Cannot Delete.

Receipt No, Date, Customer Code, Customer Name, Payment Method, Payment Reference, Remarks, Total Received, Receipt Line Item
RCT1001/20, 2020-02-04, CP, CPALL, Cash, Nothing, Paid all invoices partially, 200, [{'Invoice No': 'INT100/20', 'Invoice Date': '2020-01-03', 'Amount Paid Here': 200}]
RCT1002/20, 2020-02-06, Sam, Sam Co., Ltd., Debit Card, VISA card, Partially paid on INT100/20 and INT101/20, 9400, [{'Invoice No': 'INT100/20', 'Invoice Date': '2020-01-03', 'Amount Paid Here': 8000}, {'Invoice No': 'INT101/20', 'Invoice Date': '2020-01-04', 'Amount Paid Here': 1400}]
Result of deleting 3 receipt line, first successfully, others unsuccessfully (Press a key to continue).


Invoice No, Invoice Date, Customer Name, Invoice Amount Due, Invoice Amount Reveived, Invoice Amount Not Paid
INT100/20, 2020-01-03, Sam Co., Ltd., 14980.0, 8200, 6780.0
INT101/20, 2020-01-04, CPALL, 2140.0, 1400, 740.0
Total Debt:  7520.0

Normal Termination.   Goodbye!
```