```python
# 1, Check Digit
def checkDigit(staff_number,order_number,alphabet):
    modulus = input_alphabet(alphabet)
    order_number = order_number[-6:]
    staff_number = str(staff_number)
    order_number = str(order_number)
    result = 0
    for i in range(len(staff_number)):
        result += int(staff_number[i]) * int(order_number[i])
    return modulus - (result % modulus)
```

```python
# 2, Input Alphabet
def input_alphabet(alphabet):
 # check if the alphabet is valid
    if alphabet == 'A':
        modulus = 9
    elif alphabet == 'B':
        modulus = 8
    elif alphabet == 'C':
        modulus = 7
    elif alphabet == 'D':
        modulus = 6
    else:
        print("Invalid alphabet. Enter again.")
    return modulus
```

```python
# calculate the sub total
def cal_sub_total(items):
    sub_total = 0
    for item in items:
        sub_total += item[3]
    return sub_total
```

```python
# 4, Hash Total:
# last two digital of the order number:
def hashTotal(item_numbers):
    total = 0
    for item_number in item_numbers:
        last_two_digits = int(item_number[0][-2:])
        total += last_two_digits
    return total

# 5, mall dollar:
def mallDollar(total):
    if total >= 1000:
        mall_dollar = total * 0.002
    else:
        mall_dollar = 0
    return mall_dollar
```

```python
# 3, Cost Verification Procedure
def Cost_Verification_Procedure(items, discount_1, discount_2):
    discount = 0
    sub_total = cal_sub_total(items)

    # calculate the total discount
    discount += discount_1
    discount += sub_total * discount_2

    return sub_total - discount + deliveryFee(sub_total)

def deliveryFee(sub_total):
    # calculate the delivery_fees
    if sub_total >= 500:
        delivery_fees = 0
    else: delivery_fees = 50
    return delivery_fees
```

```python
# arbitrary assigned the customer name and address with condition statement within the function
def name_address(customer_number):
    customer_info = customer_exists(customer_number)
    if customer_info != None:
        customer_number = customer_info[0]
        customer_name = customer_info[1]
        customer_address = customer_info[2]
        return [customer_number, customer_name, customer_address]
    else:
        #generate random a new customer number 6 digits, check if the cs number exist in the csv file
        while True:
            customer_number = random.randint(100000,999999)
            if customer_exists(customer_number) == None:
                print("-Cusomter Number not exist.")
                print("-Plase Eneter Customer Name and Address")
                #add new customer, save into the csv and genereate a new customer number to the customer
                customer_name = input("Please enter the customer name: ")
                customer_address = input("Please input enter customer address: ")
                print("A New Customer Number Will be Assgined.")
                break
        #save into the csv
        with open('CustomerAddress.csv', 'a', newline='') as customer:
            csv_writer = csv.writer(customer)
            csv_writer.writerow([customer_number, customer_name, customer_address])
            print("-Customer Record Saved.")
            # stop the loop and return the customer name and address
            return [customer_number,customer_name, customer_address]
```


```python
# arbitrary assigned the customer name and address with condition statement within the function
def name_address(customer_number):
    customer_info = customer_exists(customer_number)
    if customer_info != None:
        customer_number = customer_info[0]
        customer_name = customer_info[1]
        customer_address = customer_info[2]
        return [customer_number, customer_name, customer_address]
    else:
        #generate random a new customer number 6 digits, check if the cs number exist in the csv file
        while True:
            customer_number = random.randint(100000,999999)
            if customer_exists(customer_number) == None:
                print("-Cusomter Number not exist.")
                print("-Plase Eneter Customer Name and Address")
                #add new customer, save into the csv and genereate a new customer number to the customer
                customer_name = input("Please enter the customer name: ")
                customer_address = input("Please input enter customer address: ")
                print("A New Customer Number Will be Assgined.")
                break
        #save into the csv
        with open('CustomerAddress.csv', 'a', newline='') as customer:
            csv_writer = csv.writer(customer)
            csv_writer.writerow([customer_number, customer_name, customer_address])
            print("-Customer Record Saved.")
            # stop the loop and return the customer name and address
            return [customer_number,customer_name, customer_address]
```

```python
# input for each order,
def order_input():
    # Determine the order number from the order file
    with open('OrderFile.csv', encoding="utf-8-sig") as orders:
        csv_reader = csv.reader(orders)
        next(csv_reader) # skip the header
        for line in csv_reader:
            last_order_no = line[0]
    # Then, get the new order number
    alphabet = last_order_no[0]
    num_part = str(int(last_order_no[-6:]) + 1)
    if len(num_part) > 6:
        if alphabet == "Z": # If Z encountered change to A
            alphabet = chr(ord(alphabet)-25)
        else: # Next Letter in alphabet
            alphabet = chr(ord(alphabet) + 1)
        num_part = "000001"
    for zeroNum in range(6 - len(num_part)): # make sure 6 digit
        num_part = "0" + num_part
    new_order_no = alphabet + str("-") + num_part

    # Input staff number
    while True:
        try:
            staff_number = input("Enter your staff number(6 digit): ")
            if len(staff_number) == 6:
                staff_number = int(staff_number)
                break
            print("Staff number should in 6 digit. Enter again.")
        except:
            print("Please enter a number.")
```

```
alphabet = input("Enter alphabet(A, B, C, D): ")
if alphabet == 'A':
    modulus = 9
elif alphabet == 'B':
    modulus = 8
elif alphabet == 'C':
    modulus = 7
elif alphabet == 'D':
    modulus = 6
else:
    print("Invalid alphabet. Enter again.")

# Input numbers of items within the order
global ordersNum
while True:
    try:
        numberOFItems = input("\nEnter number of items within this order (or 'q' to quit): ")
        numberOFItems = int(numberOFItems)
        if numberOFItems > 9:
            if ordersNum == 15:
                print("-The number of orders is 15.")
                print("-You are not allow to create another order.")
                print("-Enter number of items within 9 again.")
            else:
                print("-The number of items more than 9.")
                print("-One order can only have at most 9 items")
                print("-Another order will be created.")
                numberOFItems = 9
                ordersNum += 1
                break
        elif numberOFItems <= 0:
            print("The number of items should not be 0 or less than 0. Enter again.")
        else: break
    except:
        print("Please enter a number.")
```

```
    # Input item number, quantity and price
    item_list = [] # create a list, for recording item in
the order
    # fill item_list
    while True:
        if len(item_list) == numberOFItems:
            break
        item_number = input("\nEnter an item number (or
'q' to quit): ")
        isExist = False
        if item_number == 'q':
            # if the number of item is less than number
of items,
            # then ask the user to confirm for ending
input the order
            confirm = input("\nConfirm for ending this
order? (y/n)  ")
            if confirm == 'y':
                break
```

```
        # check if the item exist
        with open('ItemFile.csv', encoding="utf-8-sig")
as ItemFile:
            csv_reader = csv.reader(ItemFile)
            next(csv_reader) # skip the header
            for line in csv_reader:
                if item_number == line[0]:
                    while True:
                        try:
                            quantity_number =
input("Enter the quantity number of the item: ")
                            quantity_number =
int(quantity_number)
                            if quantity_number <= 0:
                                print("Quantity number
should not be 0 or negative.")
                                continue
                        except:
                            print("Invalid number.")
                        else:
                            break
                    for item in item_list:
                        if item[0] == line[0]:
                            item[2] += quantity_number
                            item[3] = item[2] *
float(line[2])
                            break
                    else:
                        item_list.append([line[0],
line[1], quantity_number, quantity_number *
float(line[2])])
                    isExist = True
                    break
            if not isExist:
                print("Failed to find the item.")

    # calculate the sub-total
```

```
    sub_total = cal_sub_total(item_list)

    # input the discount
    while True:
        try:
            discount_1 = input("\nEnter the 1st discount
of the order in dollar($) (if no, enter 0): ")
            discount_1 = float(discount_1)
            if discount_1 / sub_total >= 1 / 100:
                print("Actual amount of discount can only
be less than 1% of the sub-total.")
            elif discount_1 < 0:
                print("Discount can not be negative.")
            else: break
        except:
            print("Invalid amount of discount. Enter
again.")
    while True:
        try:
            discount_2 = input("\nEnter the 2nd discount
of the order in percentage(%) (0 to 5%): ")
            discount_2 = float(discount_2) / 100
            if discount_2 < 0 or discount_2 > 0.05:
                print("Invalid amount of discount. Enter
again.")
            else: break
        except:
            print("Invalid amount of discount. Enter
again.")

    while True:
        try:
            #call the name_address function to get the
customer name and address
            customer_number_input = input("Please enter
the customer number: ")
            customer_info =
name_address(customer_number_input)
```

```
            break
        except:
            print("Invalid customer number. Enter
again.")


    # hash total
    global hash_total_list
    hash_total = hashTotal(item_list)
    hash_total_list.append(hash_total) # for calculate
the new hash total

    # order total payment
    total = Cost_Verification_Procedure(item_list,
discount_1, discount_2)

    return [new_order_no, staff_number, alphabet,
sub_total, total, hash_total, numberOFItems, item_list,
discount_1,
discount_2,customer_info[0],customer_info[1],customer_inf
o[2],modulus]
```

```
def invoice(order_list):
    for order in order_list:
        modulus_number = str(checkDigit(order[1],
order[0], order[2]))
        print("%-27s AI_Tone mall" % " ")
        print("INVOICE")
        print("%-30s %-30s  %-30s" % ("Invoice Date",
"Order No.", "Mall Dollar"))
        orderNo = order[0][0] + str(order[1]) +
str(order[2]) + order[0][-6:] + str(order[6]) + "(" +
modulus_number + ")"
        print("%-30s %-30s  $%-29.2f" %
(str(dt.date.today()), orderNo,
float(mallDollar(order[4]))))
```

```
        print("%-42s %-20s %-10s" % ("Description",
"Qty", "Total"))
        print("Customer Number: " + str(order[10])) #
customerNo
        for item in order[7]:
            print("%-4s%-6s %-32s %-20s %-10.2f" %
(item[0], ".", item[1], item[2], float(item[3])))
        print()
        print("%-40s %-25s  $ %-30.2f" % ("Shipping To",
"SubTotal", float(order[3])))
        print("%-40s %-25s -$ %-30.2f" % ("Customer Name:
" +str(order[11]) , "VIP", float(order[8])))
        print("%-40s %-25s -$ %-30.2f" % ("Customer
Address: " + str(order[12]), "VIPDAY95",
float(order[9]*order[3])))
        delivery_fee = deliveryFee(order[3])
        if deliveryFee(order[3]) == 0:
            print("%-40s %-25s  %-30s" % (" ", "Delivery
Fee", "FREE"))
        else:
            print("%-40s %-25s  $ %-30.2f" % (" ",
"Delivery Fee", float(delivery_fee)))
        print("%-40s %-25s  $ %-30.2f" % (" ", "Total",
float(order[4])))
        print("\n")
```

```
# 6, Output audit file
def audited_format_file(order_list):
    with open('audited_file.txt', 'w') as f:
        # write total number of orders
        f.write("Number_of_orders: " +
str(len(order_list)) + "\n")
        # write total hash of all orders
        f.write("Hash_total_of_all_orders: " +
str(sum(hash_total_list)) + "\n")
        f.write("\n")
        # loop each order in the order list
        i = 0
```

```
        for order in order_list:
            i += 1
            f.write("Order " + str(i) + " details\n")
            # print order number
            f.write("Order_Number: " + str(order[0] +
"\n"))
            # print staff number
            f.write("Agency_number: " + str(order[1]) +
"\n")
            # print user 'modulus number'
            f.write("Modulus_number: " + str(order[13]) +
"\n")
            # print sub total
            f.write("Total: " + str(order[3]) + "\n")
            # print hash total
            f.write("Hash_total: " + str(order[5]) +
"\n")
            f.write("\n")

def last_order_file(order_list):
    with open('last_order_file.txt', 'w') as f:
        for order_no in order_list:
            order = order_no
        # print order number
        f.write("Order_Number: " + str(order[0] + "\n"))
        # print staff number
        f.write("Agency_number: " + str(order[1]) + "\n")
        # print user 'modulus number'
        f.write("Modulus_number: " + str(order[13]) +
"\n")
        # print sub total
        f.write("Total: " + str(order[3]) + "\n")
        # print hash total
        f.write("Hash_total: " + str(order[5]) + "\n")
        f.write("\n")
```

```
# main program
global ordersNum
```

```
global hash_total_list
order_list = []
hash_total_list = []
while True:
    try:
        ordersNum = input("Number of orders (1-15): ")
        ordersNum = int(ordersNum)
        if ordersNum >= 1 and ordersNum <= 15:
            break
        print("Range from 1 to 15. Input again.")
    except:
        print("Please enter a number.")
for i in range(ordersNum):
    for symbol in range(30):
        print("-", end="")
    print("\nOrder " + str(i+1) + " input,\n")
    order_list.append(order_input())
    print() # Empty line
# output()
invoice(order_list)
audited_format_file(order_list)
last_order_file(order_list)
```