



# **THE CLASSIC CUPCAKE SHOP**

online order system

# Main Menu

---

Use “art” package for text  
converting to ASCII art fancy

```
=====
The Online Cupcake
=====
```

- ```
1. Show Menu
2. Go to Cart
3. Checkout
4. Show purchase history
5. Exit the program
```

```
Please select the option(1-5): █
```

User can input the menu no. (1,2,3,4) to go to the sub menu  
or 5 to exit the program.

1. List the stock items on the screen.
2. List the items in the shopping cart on the screen.
3. Print the receipt on the screen.
4. Show the purchase history and listed on the screen.
5. Exit the program

## Sub Menu 1. Stock Items

# Cupcake Menu

|   | name                       | price | qty |
|---|----------------------------|-------|-----|
| 1 | APPLE CRUMBLE CUPCAKE      | 5.5   | 19  |
| 2 | BANANA BREAD LATTE CUPCAKE | 3.5   | 0   |
| 3 | CHOC VANILLA CUPCAKE       | 4.5   | 40  |
| 4 | RED VELVET CUPCAKE         | 8.5   | 13  |
| 5 | ROCKY ROAD CUPCAKE         | 7.5   | 23  |
| 6 | STRAWBERRY CUPCAKE         | 6.0   | 10  |
| 7 | MIX CHRISTMAS BOX          | 36.0  | 8   |

-----  
Monday Special, APPLE CRUMBLE CUPCAKE 20% off  
-----

Please input Cupcake item number (1,2,3...) to purchase the item or enter "m" back to main menu: █

User can input the item no. (1-7) to purchase the item or input "m" back to main menu.

- Once the user input the correct item no., the item will add into a shopping cart list. The quantity number in the stock item list will decrease by 1
- If user input is incorrect, the program will prompt warning message let user input the no. again
  1. The item no. is not in the list
  2. The item quantity is 0, means no stock

## Sub Menu 1. Stock Items

# Cupcake Menu

|   | name                       | price | qty |
|---|----------------------------|-------|-----|
| 1 | APPLE CRUMBLE CUPCAKE      | 5.5   | 19  |
| 2 | BANANA BREAD LATTE CUPCAKE | 3.5   | 0   |
| 3 | CHOC VANILLA CUPCAKE       | 4.5   | 40  |
| 4 | RED VELVET CUPCAKE         | 8.5   | 13  |
| 5 | ROCKY ROAD CUPCAKE         | 7.5   | 23  |
| 6 | STRAWBERRY CUPCAKE         | 6.0   | 10  |
| 7 | MIX CHRISTMAS BOX          | 36.0  | 8   |

-----  
Monday Special, APPLE CRUMBLE CUPCAKE 20% off  
-----

Please input Cupcake item number (1,2,3...) to purchase the item or enter "m" back to main menu: █

Function to list today's special items.

The special items will change everyday based on the day of the week (e.g., Monday is item 1, Tuesday is item 2)

If you purchase the special item, the price will be 20% off when you do the checkout

## Sub Menu 2. Shopping cart items

---

```
Items in Cart
```

|   | name                       | price | qty |
|---|----------------------------|-------|-----|
| 1 | APPLE CRUMBLE CUPCAKE      | 5.5   | 2   |
| 2 | BANANA BREAD LATTE CUPCAKE | 3.5   | 2   |
| 7 | MIX CHRISTMAS BOX          | 36.0  | 1   |
| 4 | RED VELVET CUPCAKE         | 8.5   | 1   |

Input item number to delete the item from your cart or enter m to back to main menu: █

User can input the item no. to delete the item from shopping cart or input “m” back to main menu.

- Once the user input the correct item no., the item will delete from this shopping cart list. At the same time the quantity number in the stock item list will increase by 1
- If user input is incorrect, the program will prompt warning message let user input the no. again (e.g., The item no. is not in the list)

## Sub Menu 2. Shopping cart items

---



Items in Cart

You don't have any items in the shopping cart, press return back to main menu....

If you don't have any items in the shopping list, when you check the shopping cart, the app will let you know nothing in the cart.

This will happen under below situations.

1. User hasn't purchase any items.
2. User delete all the items in the shopping cart.
3. After user do the checkout, all the items in user's shopping cart will be removed.

## Sub Menu 3. Checkout

Receipt

Date: 2022-12-12 10:48:08.334548 Monday

You have purchased 2 APPLE CRUMBLE CUPCAKE and price is 4.4 each

Monday Special, APPLE CRUMBLE CUPCAKE is 20% off

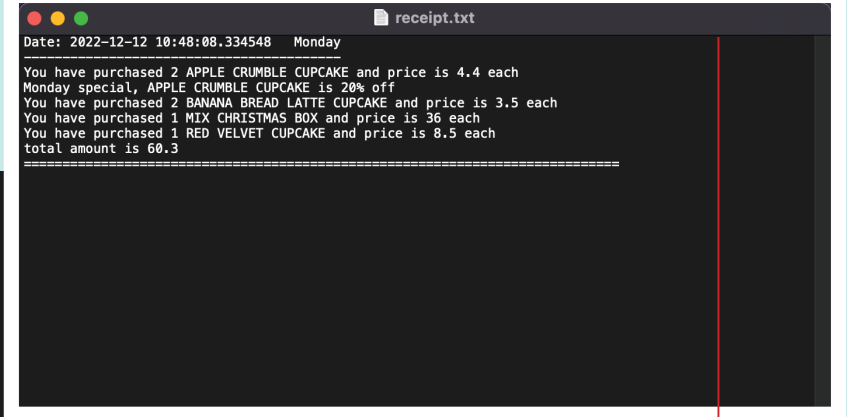
You have purchased 2 BANANA BREAD LATTE CUPCAKE and price is 3.5 each

You have purchased 1 MIX CHRISTMAS BOX and price is 36 each

You have purchased 1 RED VELVET CUPCAKE and price is 8.5 each

total amount is 60.3

Press return to continue.....



Checkout menu will display all the item info on the screen and calculate the total amount

It is including the date of purchase as well

The Special item will show the new price after the calculation

The receipt will save in a txt file if user want to check the purchase history

## Sub Menu 3. Checkout

---



```
Receipt
```

```
There is no item in cart.....  
Press return to continue.....█
```

If you don't have any items in the shopping list when you select the checkout option, the app will let the user know they can't do checkout cause nothing in the shopping cart.



## Sub Menu 4. Show purchase history

---

Purchase History

Date: 2022-12-12 10:48:08.334548    Monday

-----

You have purchased 2 APPLE CRUMBLE CUPCAKE and price is 4.4 each

Monday special, APPLE CRUMBLE CUPCAKE is 20% off

You have purchased 2 BANANA BREAD LATTE CUPCAKE and price is 3.5 each

You have purchased 1 MIX CHRISTMAS BOX and price is 36 each

You have purchased 1 RED VELVET CUPCAKE and price is 8.5 each

total amount is 60.3

=====

Date: 2022-12-12 15:15:51.893902    Monday

-----

You have purchased 1 ROCKY ROAD CUPCAKE and price is 7.5 each

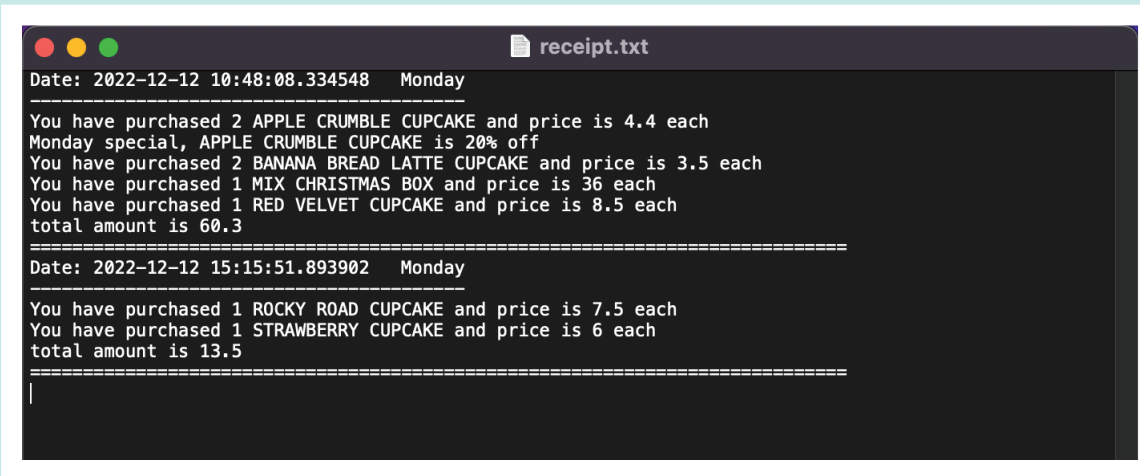
## Sub Menu 4. Show purchase history

---

```
You have purchased 1 STRAWBERRY CUPCAKE and price is 6 each  
total amount is 13.5
```

```
=====
```

```
Press return to continue.....█
```



```
receipt.txt  
Date: 2022-12-12 10:48:08.334548 Monday  
=====  
You have purchased 2 APPLE CRUMBLE CUPCAKE and price is 4.4 each  
Monday special, APPLE CRUMBLE CUPCAKE is 20% off  
You have purchased 2 BANANA BREAD LATTE CUPCAKE and price is 3.5 each  
You have purchased 1 MIX CHRISTMAS BOX and price is 36 each  
You have purchased 1 RED VELVET CUPCAKE and price is 8.5 each  
total amount is 60.3  
=====  
Date: 2022-12-12 15:15:51.893902 Monday  
=====  
You have purchased 1 ROCKY ROAD CUPCAKE and price is 7.5 each  
You have purchased 1 STRAWBERRY CUPCAKE and price is 6 each  
total amount is 13.5  
=====
```

When the user select the show purchase history option, the program will read the receipt txt file and display all the receipts on the screen.

This is the txt file for program to read from.

## Sub Menu 5. Exit the program

---

```
Thanks for your purchase  
See you next time!  
(venv) Eddys-MBP:src eddyzhou$
```

When user select the option 5, the program will exit

After option 5 selected, the program still doing something at the backend.

1. It will check the shopping cart. If the shopping cart is not empty, it will read the items info and save it in a txt file for later if user login again, they will find the shopping cart items are still there.
2. If the shopping cart is empty, the program will delete the txt file used for saving shopping cart items.

# Code challenge 1.

```
def buy_item(item_no):
    if original_dict[item_no]["qty"] == 0:
        print("Item not in stock.....Please select other items")
    else:
        if item_no in cart_dict:
            cart_dict[item_no]["qty"] += 1
            original_dict[item_no]["qty"] -= 1
            print(f'1 {cart_dict[item_no]["name"]} has been added to your cart.')
        else:
            cart_dict[item_no] = original_dict[item_no].copy()
            cart_dict[item_no]["qty"] = 1
            original_dict[item_no]["qty"] -= 1
            print(f'1 {cart_dict[item_no]["name"]} has been added to your cart.')
```

```
if select in original_dict: # check if the user input exist in stock items dictionary key value
    system("clear")
    tprint("Cupecake Menu")
    buy_item(select) # add item into shopping cart and decrease the quantity of that item in stock
    input("Press return to continue....")
    continue
elif select == "m":
    break
else:
    system("clear")
    tprint("Cupecake Menu")
    print("This item no does not exist, please input correct item no...")
    input("Press return to continue....")
```

The most challenge of this program is you need to think about every possibility of the user input. Cause 90% of the logical and conditional operation will depend on the result of user input.

Like this purchase items function in “show the items” menu. User can add the item into shopping cart by input the item no. But if the input is wrong, sometimes the whole program will crash. Here is all the input possibility.

1. If the quantity of this item in the stock list is 0. (No item in stock, can't purchase)
2. If the item's no. is already in the shopping cart, that means you want to purchase again, we just need to change the qty value instead of add same item twice.
3. If the item's no. is not in the shopping cart, but it's a valid item no. in the stock list, then add item into shopping cart list
4. Once you added the selected item into the shopping cart list, the original quantity of this item in the stock list has to decrease
5. If the input is not valid, string or number (number is not in stock list), will prompt a warning message ask user to input again

## Code challenge 2.

---

```
with open("original.txt") as f:
    data = f.read()
original_dict = json.loads(data)          # read the stock items from txt file and assign to variable original_dict as a dictionary

if os.path.exists("cart.txt"):            # check if the shopping cart is empty or not from the last exit
    with open("cart.txt") as f:
        cart_data = f.read()
        cart_dict = json.loads(cart_data)
else:
    cart_dict = {}

if cart_dict != {}:                       # adjust the quantity of stock items based on the existing shopping cart items
    for key in cart_dict:
        original_dict[key]["qty"] = original_dict[key]["qty"] - cart_dict[key]["qty"]
```

The second challenge will be read the correct file at the beginning of the program.

If the user added some items in the shopping cart during the last login. There will be a shopping cart txt file exist in the folder. So every time at the beginning of the program, you need to check the folder and reload the data from that file.

And of course the original quantity of items will change depend on this cart.txt file.

# Code challenge 3.

```
from datetime import datetime

class item:
    def __init__(self, date, item): # initial the class
        self.date = date
        self.item = item

    def add_history(self, date_file, history_dict): # display and save the date of purchase and receipt
        total_amount = 0
        f = open("receipt.txt", "a") # append new receipt in txt file
        dt = datetime.now()
        print(f"Date: {dt} {dt.strftime('%A')}\n")
        f.write(f"Date: {dt} {dt.strftime('%A')}\n")
        print("-----\n") # print on the screen
        f.write("-----\n") # save in txt file
        for i in history_dict: # loop all the items in shopping cart when checkout and calculate the total amount
            if i == date_file[dt.strftime('%A')]: # check each item if it's on discount or not based on today's special
                total_amount = total_amount + history_dict[i]['qty'] * (history_dict[i]['price']*(1-0.2))
                special_price = round(history_dict[i]['price']*(1-0.2), 2)
                print(f"You have purchased {history_dict[i]['qty']} {history_dict[i]['name']} and price is {special_price} each\n")
                f.write(f"You have purchased {history_dict[i]['qty']} {history_dict[i]['name']} and price is {special_price} each\n")
                print(f"{dt.strftime('%A')} Special, {history_dict[i]['name']} is 20% off\n")
                f.write(f"{dt.strftime('%A')} special, {history_dict[i]['name']} is 20% off\n")
            else:
                total_amount = total_amount + history_dict[i]['qty'] * history_dict[i]['price']
                print(f"You have purchased {history_dict[i]['qty']} {history_dict[i]['name']} and price is {history_dict[i]['price']} each\n")
                f.write(f"You have purchased {history_dict[i]['qty']} {history_dict[i]['name']} and price is {history_dict[i]['price']} each\n")

        print(f"total amount is {round(total_amount, 2)}\n")
        f.write(f"total amount is {round(total_amount, 2)}\n")
        print("=====\n")
        f.write("=====\n")
        f.close()
```

```
date = {"Monday": "1",
        "Tuesday": "2",
        "Wednesday": "3",
        "Thursday": "4",
        "Friday": "5",
        "Saturday": "6",
        "Sunday": "7"
}
```

```
original.txt
{
  "1": {
    "name": "APPLE CRUMBLE CUPTAKE",
    "price": 5.5,
    "qty": 20
  },
  "2": {
    "name": "BANANA BREAD LATTE CUPTAKE",
    "price": 3.5,
    "qty": 2
  },
  "3": {
    "name": "CHOC VANILLA CUPTAKE",
    "price": 4.5,
    "qty": 40
  },
  "4": {
    "name": "RED VELVET CUPTAKE",
    "price": 8.5,
    "qty": 13
  },
  "5": {
    "name": "ROCKY ROAD CUPTAKE",
    "price": 7.5,
    "qty": 23
  },
  "6": {
    "name": "STRAWBERRY CUPTAKE",
    "price": 6,
    "qty": 10
  },
  "7": {
    "name": "MIX CHRISTMAS BOX",
    "price": 36,
    "qty": 8
  }
}
```

The third challenge is how to connect the discount items with the day of the week

We have a list of discount items (Monday discount item is item no. 1, Tuesday discount item is item no. 2 ....)

We use for loop to compare each items in the shopping cart with the discount items list. If they are matching, the price will calculate based on the discount rate. And added to the total amount in the end.