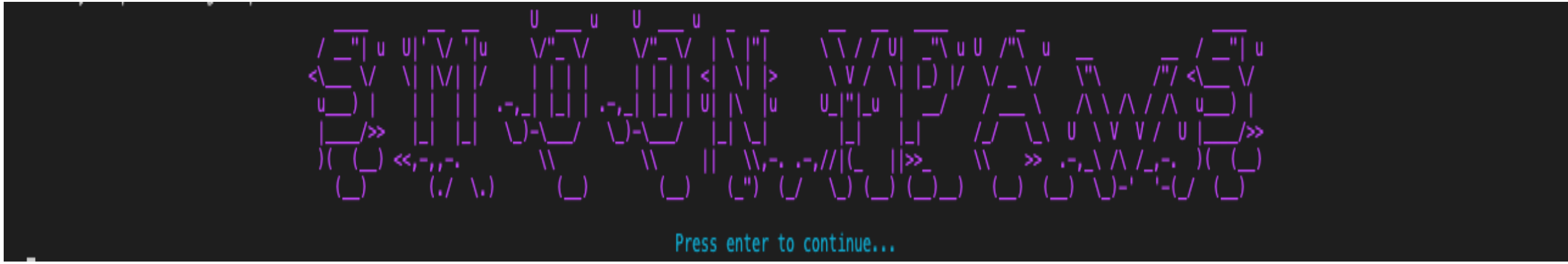
Sticker store ordering system

Smoonypaws



Walkthrough

- Users have to enter the correct password to start the system (they could set up their password by execute a bash script)
- If they enter a wrong password, they have chance to enter again.
- Select one of these options to proceed

```
Enter your password:password

                                   Press enter to continue...

Options
> 1: Collect information & Order stickers
  2: Display existing customer information
  3: Check customer order history
  4: Add update menu items
  5: Add sold out menu items
  6: Display products
  0: Exit the system
```

Collect customer information and order

- Collect each customer's name
- Ask users what customer wants to order and the quantity of the selected sticker. Quantity must be an positive integer and greater than 0.
- Ask users whether the customer would like to order antoher stickers.
- If users do not enter Y or N, they have chance to enter again.
- Print a receipt for the customer with time.

```
Enter the name of the customer:
Alex
Enter the sticker name[Enter a valid sticker only, e.g. Yum Yum Hana]:
Yum Yum Hana
Enter the sticker quantity[Enter a positive integer only, e.g. 1, 2, 3]:
10
Want to order another sticker? Y or N
N
Ordering time                    23 September 2022, Friday 18:33:40 PM

*****************************************************************
Receipt of Customer Alex
*****************************************************************
Yum Yum Hana:                                        10.0(AUD) x 10
Discount:                                                      -10%
Total Cost:                                              90.0(AUD)
*****************************************************************
Printing time                    23 September 2022, Friday 18:33:51 PM

Press enter to continue...
```

Allow customers to join rewards membership

- Each customer will be automatically added to the customer list
- The system will ask users whether the customer would like to become rewards customer
- If users do not enter Y or N, they have chance to enter again.

```
The customer is not in the rewards program. Does the customer want to join the rewards program
[Enter Y/N]?
Y
Successfully add the customer to the rewards program
```

Displaying existing customers

- Users could select number 1 or 2 to display either non-rewards customers or rewards customer.
- If they enter something else instead of 1 or 2, they have chance to enter again.

```
                                                    You have selected to display the customer information
Please select customer group: 1 for non-rewards customer, 2 for rewards customer
2
   Customer Information
```

Name	Rewards Customer
Alex	✅

```
                                                    Press enter to continue...
```

Displaying customer order history

- Users could select a customer and display that customer's order history. Each order will be displayed on a single line.
- If users enter a name that is not an existing customer, they have chance to enter again.
- if that customer has no order history, a message will be displayed on the terminal.

```
You have selected to display the customer order history

Please enter a customer name for checking order history:
Alex
Customer order history
            This is the order history of Alex
```

Orders	Stickers	Total Cost
1	10xYum Yum Hana|10xWinter Vibes|	90.0

```
Press enter to continue...
```

Display all products

- Display all products, includes their name, price and whether they have been sold out.

```
You have selected to display all products

This is the product list
```

Products	Price	Sold out
Yum Yum Hana	10.0	✅
Winter Vibes	5.0	❌
Spring Vibes	5.0	❌
Summer Vibes	5.0	❌
Autumn Vibes	5.0	❌

```
Press enter to continue...
```

Add and update menu

- Users could add new items to the menu or update the price of existing menu items.
- Users could also mark items as sold out.
- If users enter invalid input, that input will be ignored.
- If users use wrong format, like they only enters a string instead of key:value pair. They have chance to enter again.

```
                                                    You have selected to add and update men
Please enter the new stickers and prices with the following format: sticker_1 : price_1, sticker_2: price_2
Yum Yum Hana:10.0, Winter Vibes:-5.0
The price entered for Winter Vibes is not valid
                                                    Press enter to continue...
```

```
                                                    You
Please add sold out stickers
Yum Yum Hana
```

Option menu

- Users could choose which function they would like to proceed
- Once that function finished, they will return to the option menu

```python
def welcome_page():
    options = ['1: Collect information & Order stickers',
     '2: Display existing customer information',
     '3: Check customer order history',
     '4: Add update menu items',
     '5: Add sold out menu items',
     '6: Display products',
     '0: Exit the system']
    terminal_menu = TerminalMenu(options, title='Options')
    menu_entry_index = terminal_menu.show()
    match menu_entry_index:
        case 0:
            collect_info_and_order()
            press_enter.color_input()
            welcome_page()
        case 1:
            customer_info()
            press_enter.color_input()
            welcome_page()
        case 2:
            check_history()
            press_enter.color_input()
            welcome_page()
        case 3:
            add_update()
            press_enter.color_input()
            welcome_page()
        case 4:
            add_sold_out_stickers()
            press_enter.color_input()
            welcome_page()
        case 5:
            display_products()
            press_enter.color_input()
            welcome_page()
        case 6:
            clearing.clear()
            print('Bye')
            sys.exit()
```

Loops

```
sys.stdout.write('Enter the sticker name'
                 '[Enter a valid sticker only, e.g. Yum Yum Hana]:\n')
sticker = str(sys.stdin.readline().strip())
# While loop to check whether the input sticker name is in the Menu.
# If not in the Menu, then the while loop will keep looping.
while sticker not in menu:
    sys.stdout.write('Please enter a valid sticker name:'+ '\n')
    sticker = str(sys.stdin.readline().strip())
```

```
while not valid:
    sys.stdout.write('Want to order another sticker? ' + 'Y or N' + '\n')
    another_sticker = str(sys.stdin.readline().strip())

    # If the answer is not N, then the loop will repeat.
    if another_sticker == 'N':
        valid = True
        return valid
    elif another_sticker == 'Y':
        sys.stdout.write('Enter a valid sticker name' + '\n')
        new_sticker_name = str(sys.stdin.readline().strip())
        while new_sticker_name not in menu:
            # Read the new input
            new_sticker_name = str(input('Please enter a valid sticker name:\n').strip())
        sticker = new_sticker_name
        self.record_quantity()
```

- Use loops to check users input. If they enter an invalid input, the loop will repeat and ask them to enter again.
- Used for majority of my features

Error handling

```python
while quantity_is_float:
    sys.stdout.write('Enter the sticker quantity'
                     + '[Enter a positive integer only, e.g. 1, 2, 3]:\n')
    quantity = (sys.stdin.readline().strip())
    # Use try-except to exclude float, 0, negative and input that not a nu
    try:
        quantity_1 = int(quantity)
        if quantity_1 > 0:
            quantity_is_float = False
    except ValueError:
        quantity_is_float = True
```

- ValueError to handle invalid input such as string.
- IndexError use to handle input with wrong format. Like enter a string instead of key:value pair.

```python
try:
    search_menu()
    string = str(input('Please enter the new stickers and prices with the following format: '
                       + 'sticker_1 : price_1, sticker_2: price_2\n').strip())
    string_list = string.split(',')
    new_menu_list = []

    # For loop to update the stickers and prices
    for x in string_list:  #obtain each sticker and their cost as a single string
        change_list = x.split(':')
        # Remove space for each string, then form a list.
        # Later use for loop to go though the list and update the menu and price.
        for i in change_list:
            new_sticker = (i.strip())
            new_menu_list.append(new_sticker)
        # Use try-except to exclude input that is 0, negative or not a number.
        try:
            new_price = float(new_menu_list[1])
            if new_price > 0:
                menu[new_menu_list[0]] = new_price
            else:
                print('The price entered for' + ' ' + str(new_menu_list[0] +' ' + 'is not valid'))
        except ValueError:
            print('The price entered for' + ' ' + str(new_menu_list[0] +' ' + 'is not valid'))
        new_menu_list = []

    with open('Menu.json', 'w', encoding='utf8') as menu_list:
        json.dump(menu, menu_list)

    return menu

except IndexError:
    add_update()
```

Use json file to store data

- Rewards_customer_list.json
- Sold_out_list.json
- Menu.json
- Customer_list.json
- Customer_order_history.json M

- Use json file to store data.
- System could trieve data from json files and update it.

```python
def search_customer(self):
    '''Retrieve customer information
    '''
    global customer_list
    with open('Customer_list.json', 'rb') as customer:
        customer_list = json.load(customer)

    return customer_list


def add_customer(self):
    '''Add customer to the customer list
    '''
    self.search_customer()
    customer_list.append(self.name)
    with open('Customer_list.json', 'w', encoding='utf8') as customer:
        json.dump(customer_list, customer)
```

Command Line Arguments

- Create severak bash scripts to allow users to run the program from the terminal.

start_system.sh

order_only.sh

display_info_only.sh

full_options.sh

help.sh

src > start_system.sh

```bash
#!/bin/bash
if [[ -x "$(command -v python3)" ]]
then
    pyv="$(python3 -V 2>&1)"
    if [[ $pyv == "Python 3"* ]]
    then
        python3 password.py
    else
        echo "You've got the wrong version of python, sort it out!
        To install Python, check out https://installpython3.com/ ">&2
    fi
else
    echo "You don't have python, go get it!
    To install Python, check out https://installpython3.com/ " >&2
fi
```

Summary

- Challenges:

-How to link all options together

-How to use json file to store data and retrieve data from these files

-Unit testing

- Reflection:

-Next time could add more classes, such as item class, operation class, etc. Hope to make the code more simple.