COAL BOOK_SHOP PROJECT REPORT

Submitted by: Abdullah Noor

Submitted to: Sir Ashraf

Roll no. BCSF22M025

Date: 02-06-2024

=> Online Book Shop Project Report

Project Overview:

The Online Book Shop project is designed as an online portal where users can create an account, log in, browse through different categories of books, and purchase them. The system is developed using assembly language and aims to provide a seamless experience for users to buy books from various genres.

Features:

1- User Authentication:

- **Create Account:** Users can create an account by entering their email and password.
- **Login:** Existing users can log in using their credentials. The system verifies the email and password before granting access.
- **Logout:** Users can log out of their account at any time.

2- Book Categories:

Once logged in, users can browse through different categories of books:

- English Novels
- Urdu Novels
- Islamic Books

3- Book Selection and Purchase:

- **Book List Display:** Each category contains a list of books with their respective prices.
- **Book Selection:** Users can select a book from the list and specify the quantity they want to purchase.
- **Price Calculation:** The total price is calculated based on the selected quantity and the price of the book.

- **Continue Shopping or Exit:** Users can choose to buy more books or exit the portal.

User Interface:

1- Welcome Screen:

The welcome screen displays the name of the shop and provides options for creating an account or logging in.

2- Main Menu:

After logging in, the main menu displays the categories of books available. Users can choose a category to view the list of books.

3- Book Lists:

Each category displays a list of books along with their prices. Users can select a book to view more details and proceed with the purchase.

4- Input and Validation:

- The system prompts the user to enter their email and password when creating an account and logging in.
- The email and password are compared with stored values to ensure correct login credentials.
- If incorrect credentials are entered, appropriate error messages are displayed.

Data Storage:

The system uses string variables to store user credentials and book details. The book prices and lists are hardcoded in the program for simplicity.

Program Flow:

Start:

- Display the welcome screen and options to create an account or log in.
- Prompt the user to choose an option.

Account Creation:

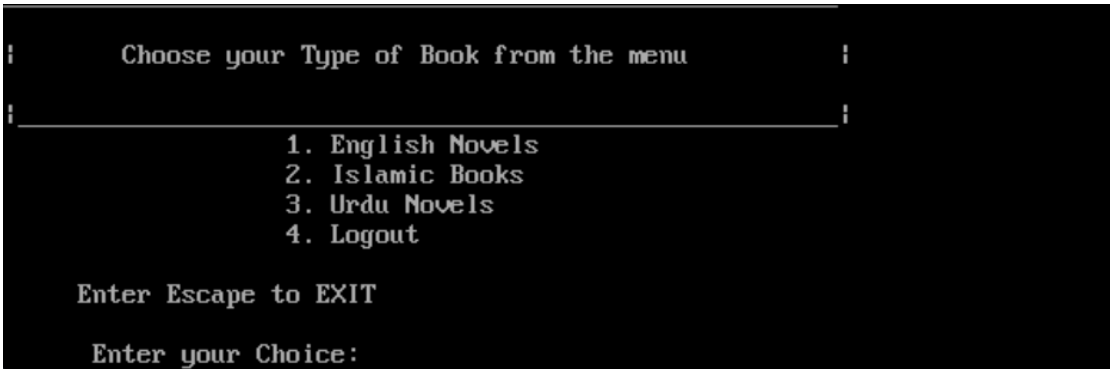
- Prompt the user to enter their email and password.
- Store the credentials and display a success message.
- Redirect to the welcome screen.

Login:

- Prompt the user to enter their email and password.
- Validate the credentials.
- If valid, display the main menu.
- If invalid, display an error message and prompt to re-enter credentials.

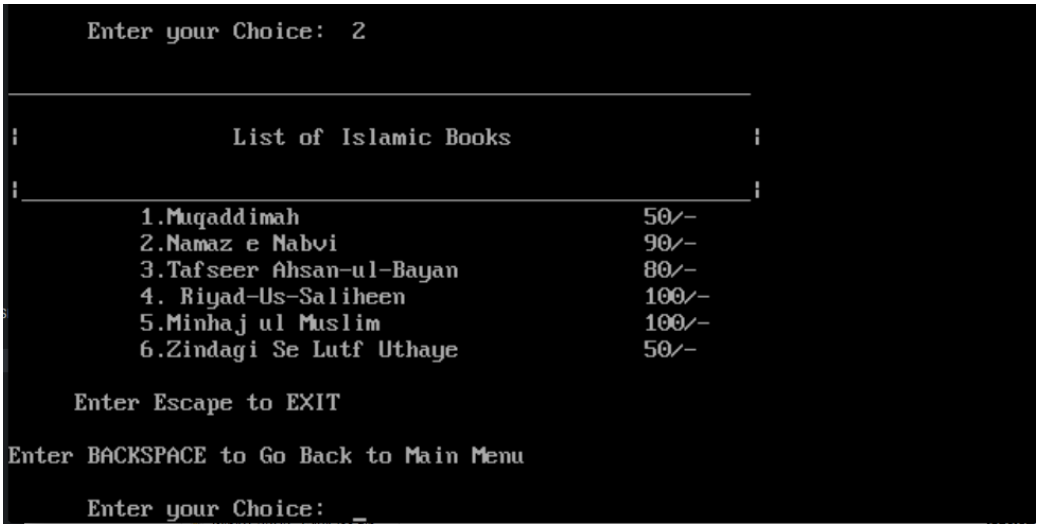
Main Menu:

- Display the categories of books.
- Prompt the user to choose a category.



Book List:

- Display the list of books in the chosen category followed by price.
- Prompt the user to choose a book.



Book Selection:

- Prompt the user to enter the quantity.
- Calculate and display the total price.

Continue Shopping:

- If the user chooses to continue, redirect to the main menu.
- If the user chooses to exit, press escape and exit.

Code Analysis:

1- MACRO:

```
PRINT_STR MACRO STR
        LEA DX , STR
        MOV AH , 9
        INT 21H
ENDM
```

- This macro is called **PRINT_STR** and it takes one parameter STR, which is the string to be printed.
- **PRINT_STR MACRO STR:** This line defines the macro PRINT_STR with one parameter STR.
- So, when you use this macro PRINT_STR, you pass a string as an argument, and it will print that string to the console.

2- DATA SEGMENT:

D1: This label represents a string that includes newline and carriage return characters (10, 13) repeated twice, followed by a series of hyphens (-) and a dollar sign ($) which indicates the end of the string.

D2 to D6: These labels represent strings that include formatted text with newlines and carriage returns at the beginning, and the dollar sign at the end.

D7: Similar to D1, this label represents a string with newlines, carriage returns, and hyphens.

MS000 to MS00: These labels represent strings used to format the welcome message and other sections of the display.

MS2 to MS5: These labels represent strings for account creation and login prompts.

MS111, MS6, and MS011: These labels represent strings indicating account creation success and related messages.

MSG1 to MSG4: These labels represent strings prompting the user for account-related decisions.

MSG5 and MSG6: These labels represent error messages for incorrect email or password.

MS211, MSG7, and MS022: These labels represent strings indicating successful login and related messages.

MS7 to MSG9: These labels represent strings prompting the user to interact with the book list and order process.

MS411, MS9, and MS044: These labels represent strings prompting the user to choose a type of book.

MS10 to MS13: These labels represent menu options for selecting different types of books.

MS511, MS14, and MS055: These labels represent headers and footers for the list of English novels.

MS15 to MS20: These labels represent the list of English novels with their prices.

MS611, MS21, and MS066: These labels represent headers and footers for the list of Islamic books.

MS22 to MS27: These labels represent the list of Islamic books with their prices.

MS711, MS28, and MS077: These labels represent headers and footers for the list of Urdu novels.

MS29 to MS33: These labels represent the list of Urdu novels with their prices.

MS34 and MS35: These labels represent error messages for invalid entries.

MS36 to MS38, and MS311 and MS033: These labels represent prompts for entering orders, quantities, and displaying the total price.

QUANTITY and ITERATE: These labels define single-byte variables. QUANTITY is used to store the quantity of books ordered, and ITERATE is used to store a loop counter initialized to the character '0'.

MS39 and MS40: These labels represent strings prompting the user to navigate back to the main menu or exit the application.

STR_NAME, STR_PASS, STR_NAME1, and STR_PASS1: These labels define 50-byte arrays (each byte initialized to $). They are used to store the user's name and password.

FIRST and SECOND: These labels define single-byte variables.

suj and TEMP: These labels define double-word variables (two bytes each).

Summary:

The data segment contains numerous labels representing strings and variables used throughout the assembly program. These strings are mainly for displaying messages, menus, and prompts to the user, while the variables store input data and control the program flow.

3- CODE SEGMENT:

MAIN PROC: Begins the main procedure.

MOV AX, @DATA: Loads the data segment address into the AX register.

MOV DS, AX: Sets the DS (Data Segment) register to the value in AX, initializing the data segment.

MOV ES, AX: Sets the ES (Extra Segment) register to the value in AX, also initializing the extra segment.

D1 to D7: These lines call the PRINT_STR macro to print various strings defined earlier (D1 through D7) which together form the welcome screen.

=> Main Menu:

LABELSTART: Label which starts the main menu in which there are menu options and the user has to choose.

MS000 to MS8: These lines print the main menu options and prompts using the PRINT_STR macro.

MOV AH, 1: This block of code takes user input to select an option from the main menu.

CMP AL, '1' to JMP LABELSTART: These instructions compare the input received from the user and take appropriate actions based on the selection.

EXITT1: This block of code handles program termination if the user chooses to exit.

=> Creating Account:

ACCOUNT: This Label indicates the start of the account creation process.

PRINT_STR MS4: Displays the prompt for the user to enter their email or name.

LEA DI , STR_PASS and LEA SI , STR_NAME: These lines initialize pointers to the memory locations where the email and password strings will be stored.

A1: The instructions in label A1 reads a character from the input for name. These instructions check if the Enter key was pressed and if not, store the character in the email string and continue reading characters.

A21-PRINT_STR MS5: This line prints a prompt for entering the password in A21 Label.

JMP A2: This instruction jumps to the next block of code to read the password string.

A2: The instructions in the A2 Label reads a character from the input without echoing it to the screen.These instructions check if the Enter key was pressed. If not, the character is stored in the password string, and '*' is displayed on the screen.

SUCCESS: The instructions in the SUCCESS Label show successful completion of the account creation process.

SUCCESS-PRINT_STR MS6: These lines print a success message and after that execute (**JMP LABELSTART**)--return to the main menu after a successful account creation.

=> Login:

LOGIN: The instructions in LOGIN Label print prompts for the login process and initialize the pointer BX to the memory location where the email string for login will be stored.

L1: The instructions in L1 Label indicate the beginning of a loop to take input for the email during login.(**MOV AH , 1**)This instruction reads a character from the input.

L1-(CMP AL, 13) to (JMP L1): These instructions check if the Enter key was pressed. If not, the character is stored in the email string, and the loop continues.

PUT: This label marks the continuation of input after the email has been entered.

PUT-(LEA BX,STR_NAME1) to (JMP L21): These instructions prepare the pointers for comparing the entered email with the stored email.

L21: This Label indicates the beginning of the loop to compare the entered email with the stored email.

L21-(MOV AL, [SI]) to (JMP L21): These instructions compare each character of the entered email with the stored email. If a mismatch is found, it jumps to handle incorrect input.

L211: This label marks the handling of incorrect email input.

L211-(PRINT_STR MSG5) to (JMP LOGIN): These lines print a message indicating incorrect email input and prompt for login again.

L22: This label marks the continuation of input after the email has been successfully entered.

L22-(LEA BX,STR_PASS1) to (PRINT_STR MS5): These instructions prepare for input of the password.

L2: This Label marks the beginning of the loop to take input for the password.

L2-(MOV AH, 7): This instruction reads a character from the input without echoing it to the screen.

L2-(CMP AL, 13) to (JMP L2): These instructions check if the Enter key was pressed. If not, the character is stored in the password string, and '*' is displayed on the screen.

CMP_PASS: This label marks the comparison of the entered password with the stored password.

CMP_PASS-(LEA BX, STR_PASS1) to (LEA DI, STR_PASS): These instructions prepare pointers for comparing the entered password with the stored password.

t1: This label marks the beginning of the loop to compare the entered password with the stored password.

t1-(MOV AL, [DI]) to (JMP t1): These instructions compare each character of the entered password with the stored password. If a mismatch is found, it jumps to handle incorrect input.

L2111: This label marks the handling of incorrect password input.

L2111-(PRINT_STR MSG6) to (JMP L22): These lines print a message indicating incorrect password input and prompt for password again.

PR: This label marks the successful login process.

PR-(PRINT_STR MS211) to (JMP MAIN_MENU): These lines print a success message and jump to the main menu after successful login.

=> Books Menu:

MAIN_MENU: This Label marks the beginning of the main menu section.

MAIN_MENU-(PRINT_STR MS411) to (PRINT_STR MS8): These instructions print the main menu options.

MAIN_MENU-(MOV AH, 1): This instruction reads a single character input from the user.

MAIN_MENU-(CMP AL, '1') to (JMP MAIN_MENU): These instructions compare the input character with the menu options and jump to corresponding sections based on the input. If the input doesn't match any option, it prints an error message and loops back to the main menu.

LOGOUT: This label marks the section for logging out of the program.

LOGOUT-(JMP LABELSTART): This instruction jumps back to the beginning of the program to display the welcome screen again.

ISLAMIC_BOOKS111-(JMP ISLAMIC_BOOKS11): This instruction jumps to the section for displaying Islamic books.

URDU_NOVELS111-(JMP URDU_NOVELS11): This instruction jumps to the section for displaying Urdu novels.

ENGLISH: This Label marks the beginning of the section for displaying English novels.

ENGLISH-(PRINT_STR MS511) to (PRINT_STR MS40): These instructions print the list of English novels.

ENGLISH-(MOV AH, 1): This instruction reads a single character input from the user to select a novel.

ENGLISH-(CMP AL , '1') to (JMP ENGLISH): These instructions compare the input character with the options for selecting a novel and jump to the corresponding sections based on the input. If the input doesn't match any option, it prints an error message and loops back to the English novel menu.

ISLAMIC_BOOKS11: This label marks the beginning of the section for displaying Islamic books using **JMP ISLAMIC_BOOKS**.

URDU_NOVELS11: This label marks the beginning of the section for displaying urdu novels using **JMP URDU_NOVELS1**

FIFTY111: This Label helps to Calculate Price, which Book price is 50 using **JMP FIFTY11.**

NINETY111: This Label helps to Calculate Price, which Book price is 90 using **JMP NINETY11.**

SEVENTY111: This Label helps to Calculate Price, which Book price is 70 using **JMP SEVENTY11.**

EIGHTY111: This Label helps to Calculate Price, which Book price is 80 using **JMP EIGHTY11.**

HUNDRED111: This Label helps to Calculate Price, which Book price is 100 using **JMP HUNDRED11.**

ISLAMIC_BOOKS-(PRINT_STR MS611) to (PRINT_STR MS8): These instructions print the list of Islamic books.

ISLAMIC_BOOKS-(MOV AH, 1): This instruction reads a single character input from the user to select a book.

ISLAMIC_BOOKS-(CMP AL , '1') to (JMP ISLAMIC_BOOKS): These instructions compare the input character with the options for selecting a book and jump to the corresponding sections based on the input. If the input doesn't match any option, it prints an error message and loops back to the English novel menu.

URDU_NOVELS1: This label marks the beginning of the section for displaying urdu novels using **JMP URDU_NOVELS**

FIFTY11: This Label helps to Calculate Price, which Book price is 50 using **JMP FIFTY1.**

NINETY11: This Label helps to Calculate Price, which Book price is 90 using **JMP NINETY1.**

SEVENTY11: This Label helps to Calculate Price, which Book price is 70 using **JMP SEVENTY1.**

EIGHTY11: This Label helps to Calculate Price, which Book price is 80 using **JMP EIGHTY1.**

HUNDRED11: This Label helps to Calculate Price, which Book price is 100 using **JMP HUNDRED1.**

URDU_NOVELS-(PRINT_STR MS711) to (PRINT_STR MS8): These instructions print the list of Urdu novels.

URDU_NOVELS-(MOV AH, 1): This instruction reads a single character input from the user to select a novel.

URDU_NOVELS-(CMP AL , '1') to (JMP URDU_NOVELS): These instructions compare the input character with the options for selecting a novel and jump to the corresponding sections based on the input. If the input doesn't match any option, it prints an error message and loops back to the English novel menu.

FIFTY1: This Label helps to Calculate Price, which Book price is 50 using **JMP FIFTY.**

NINETY1: This Label helps to Calculate Price, which Book price is 90 using **JMP NINETY.**

SEVENTY1: This Label helps to Calculate Price, which Book price is 70 using **JMP SEVENTY.**

EIGHTY1: This Label helps to Calculate Price, which Book price is 80 using **JMP EIGHTY.**

HUNDRED1: This Label helps to Calculate Price, which Book price is 100 using **JMP HUNDRED.**

=> Price Calculation Sections:

For 80:

EIGHTY: This Label helps to calculate and display the total price based on the selected quantity.

EIGHTY-(MOV AH, 1): Displays a prompt asking the user to enter the quantity of books.

EIGHTY-(SUB AL, 30H) to (AAM): Reads the user input, subtracts 30H to convert ASCII to hexadecimal, multiplies the result by 80, and adjusts the result using AAM.

EIGHTY-(MOV FIRST, AH) to (ADD SECOND, 30H): Divides the result into two parts, stores them in FIRST and SECOND, and converts them back to ASCII.

EIGHTY-(PRINT_STR MS311) to (JMP K): Displays the total price, prints the result, and jumps to the K label.

K-(PRINT_STR MS39) to (JMP K): Prompts the user to go back to the main menu or exit the program. If the user presses Backspace (ASCII 8), it jumps to the main menu; if the user presses Escape (ASCII 27), it exits the program.

For 50:

FIFTY: This Label helps to calculate and display the total price based on the selected quantity.

FIFTY-(MOV AH, 1): Displays a prompt asking the user to enter the quantity of books.

FIFTY-(SUB AL, 30H) to (AAM): Reads the user input, subtracts 30H to convert ASCII to hexadecimal, multiplies the result by 50, and adjusts the result using AAM.

FIFTY-(MOV FIRST, AH) to (ADD SECOND, 30H): Divides the result into two parts, stores them in FIRST and SECOND, and converts them back to ASCII.

FIFTY-(PRINT_STR MS311) to (JMP KK): Displays the total price, prints the result, and jumps to the KK label.

KK-(PRINT_STR MS39) to (JMP KK): Prompts the user to go back to the main menu or exit the program. If the user presses Backspace (ASCII 8), it jumps to the main menu; if the user presses Escape (ASCII 27), it exits the program.

For 90:

NINETY: This Label helps to calculate and display the total price based on the selected quantity.

NINETY-(MOV AH, 1): Displays a prompt asking the user to enter the quantity of books.

NINETY-(SUB AL, 30H) to (AAM): Reads the user input, subtracts 30H to convert ASCII to hexadecimal, multiplies the result by 90, and adjusts the result using AAM.

NINETY-(MOV FIRST, AH) to (ADD SECOND, 30H): Divides the result into two parts, stores them in FIRST and SECOND, and converts them back to ASCII.

NINETY-(PRINT_STR MS311) to (JMP KK1): Displays the total price, prints the result, and jumps to the KK1 label.

KK1-(PRINT_STR MS39) to (JMP KK1): Prompts the user to go back to the main menu or exit the program. If the user presses Backspace (ASCII 8), it jumps to the main menu; if the user presses Escape (ASCII 27), it exits the program.

For 100:

HUNDRED: This Label helps to calculate and display the total price based on the selected quantity.

HUNDRED-(MOV AH, 1): Displays a prompt asking the user to enter the quantity of books.

HUNDRED-(SUB AL, 30H) to (AAM): Reads the user input, subtracts 30H to convert ASCII to hexadecimal, multiplies the result by 100, and adjusts the result using AAM.

HUNDRED-(MOV FIRST, AH) to (ADD SECOND, 30H): Divides the result into two parts, stores them in FIRST and SECOND, and converts them back to ASCII.

HUNDRED-(PRINT_STR MS311) to (JMP KK2): Displays the total price, prints the result, and jumps to the KK1 label.

KK2-(PRINT_STR MS39) to (JMP KK2): Prompts the user to go back to the main menu or exit the program. If the user presses Backspace (ASCII 8), it jumps to the main menu; if the user presses Escape (ASCII 27), it exits the program.

For 70:

SEVENTY: This Label helps to calculate and display the total price based on the selected quantity.

SEVENTY-(MOV AH, 1): Displays a prompt asking the user to enter the quantity of books.

SEVENTY-(SUB AL, 30H) to (AAM): Reads the user input, subtracts 30H to convert ASCII to hexadecimal, multiplies the result by 70, and adjusts the result using AAM.

SEVENTY-(MOV FIRST, AH) to (ADD SECOND, 30H): Divides the result into two parts, stores them in FIRST and SECOND, and converts them back to ASCII.

SEVENTY-(PRINT_STR MS311) to (JMP KK3): Displays the total price, prints the result, and jumps to the KK1 label.

KK3-(PRINT_STR MS39) to (JMP KK3): Prompts the user to go back to the main menu or exit the program. If the user presses Backspace (ASCII 8), it jumps to the main menu; if the user presses Escape (ASCII 27), it exits the program.

=> Break Procedure:

```
BREAK PROC
        MOV     DX,10
        MOV     AH,2
        INT     21H
        MOV     DX,13
```

```
    MOV     AH,2
    INT     21H
    RET
BREAK ENDP
```

- MOV DX, 10: Load the ASCII value for Line Feed (LF) into DX.
- MOV AH, 2: Set AH to 2, which is the DOS function to print a character.
- MOV DX, 13: Load the ASCII value for Carriage Return (CR) into DX.
- MOV AH, 2: Set AH to 2, which is the DOS function to print a character.
- RET: Return from the BREAK procedure to the calling code.

We can use the BREAK procedure wherever a new line is required in the main program. For example:

```
    PRINT_STR MSG1
    CALL BREAK
    PRINT_STR MSG2
```

==== END OF ASSEMBLY PROGRAM ====