Assembly Program Documentation

Overview

This program is an x86 Assembly-based calculator application that provides basic arithmetic operations such as addition, subtraction, multiplication, division, negation, increment, and decrement. The program is designed to run in a DOS environment, utilizing interrupts for input/output operations.

Structure

1. Data Section

- **Purpose**: Contains messages and variables used throughout the program.
- **Key Components**:
  - **menu1**: "Select an operation (1-7): $"
  - **operation**: Message template for selected operation.
  - **result_msg**: Displays the result of calculations.
  - **THANK**: "Thank you for using our application."
  - **INVALID**: Message for invalid input.
  - **CHOOSE**: Prompts the user to select an option.
  - **num1, num2**: Prompts for input numbers.
  - **temp**: Stores temporary calculations.
  - **res**: Stores the result of the current operation.
  - **TH, T**: Constants for digit manipulation.

2. Code Section

- **Purpose**: Implements the logic and operations of the calculator.
- **Components**:
  - **MAIN**: Entry point and main loop for menu and operation handling.
  - **Arithmetic Procedures**:
    - `do_add`, `do_sub`, `do_mul`, `do_div`: Perform respective arithmetic operations.
    - `do_neg`: Negates the input number.
    - `do_inc`: Increments the input number by 1.
    - `do_dec`: Decrements the input number by 1.
  - **Input Handling**:
    - `INPUT_4DIGIT_NUM`: Reads a 4-digit number from the user.
  - **Output Handling**:
    - `PRINT_NUMBER`: Displays numbers, including handling negative values.
    - `SUCCESS_MSG`: Displays a success message along with the operation result.
  - **Error Handling**:

        - Displays an error message for invalid menu options or inputs.
    - **Exit Procedure**:
        - Displays a thank-you message and terminates the program.

Program Flow

1. **Initialization**:
    - Loads the `.DATA` segment.
    - Displays the main menu.
2. **Operation Selection**:
    - User selects an option by entering a number (1-7).
    - If input is invalid, an error message is displayed, and the program prompts again.
3. **Execution of Operations**:
    - Depending on the option selected, the corresponding arithmetic operation procedure is called.
    - Inputs are validated, results are calculated, and displayed to the user.
4. **Exit**:
    - On selecting option `7`, the program displays a thank-you message and exits.

Arithmetic Procedures

Addition (`do_add`)

- Prompts for two 4-digit numbers.
- Adds the numbers and stores the result in `res`.
- Displays the result using `SUCCESS_MSG`.

Subtraction (`do_sub`)

- Prompts for two 4-digit numbers.
- Subtracts the second number from the first.
- Handles cases where the result is negative.

Multiplication (`do_mul`)

- Prompts for two 4-digit numbers.
- Multiplies the numbers.
- Ensures non-negative results.

Division (`do_div`)

- Prompts for two 4-digit numbers.

- Divides the first number by the second.
- Handles division by zero safely.

Negation (`do_neg`)

- Negates the input number and stores the result in `res`.

Increment (`do_inc`)

- Increments the input number by 1.

Decrement (`do_dec`)

- Decrements the input number by 1.

Input Handling

- **INPUT_4DIGIT_NUM**:
  - Reads a 4-digit number from the user.
  - Validates and processes the input character by character.

Output Handling

- **PRINT_NUMBER**:
  - Converts the number into ASCII and prints it digit by digit.
  - Handles negative numbers by prepending a `-` sign.
- **SUCCESS_MSG**:
  - Displays a success message along with the computed result.

Error Handling

- Invalid menu options:
  - Displays the message "Invalid input. Please choose a different option."
  - Returns to the main menu for a valid selection.
- Division by zero:
  - Ensures safe handling and displays a failure message.

Interrupts Used

- **INT 21H**:
  - **AH = 09H**: Display a string.
  - **AH = 01H**: Accept a single character input.
  - **AH = 02H**: Display a single character.
  - **AH = 4CH**: Exit the program.

Example Workflow

1. Program starts and displays the main menu.
2. User selects `1` for addition.
3. User enters `1234` and `5678`.
4. Result (`6912`) is displayed.
5. User selects `7` to exit, and a thank-you message is displayed.