High-Level Functionality

A program designed to create a data repository and based on what each piece of data contains, it will perform several data management operations. Some of these operations include, user data entry, access to data, arithmetic functions, and basic conditional branching.

User Stories

Daniel is a student, who is looking to perform basic arithmetic calculations, so he can work on a school project.

Tabitha is a data entry clerk, who is looking to manage customer points, so they can do their job efficiently and satisfy the customer.

Use Cases

Actor: User
System Goal: Store the accumulator result in memory
Preconditions: User has finished an arithmetic operation
System stores the value in the accumulator at the specified memory address.
If the memory address is out of range, the system calls an error.
System retrieves the stored value from memory and prints it to the terminal.
Store User Input in Memory

Actor: User
System Goal: Store user input in memory
Preconditions: User wants to store a value in memory
System prompts the user for input and validates it.
System stores the validated input in the specified memory address.
Add or Subtract Number in Memory from Accumulator

Actor: User
System Goal: Perform addition or subtraction between memory and accumulator
Preconditions: User wants to add or subtract a value in memory from the accumulator
System retrieves the value from memory at the specified address.
System performs the addition or subtraction operation between the value in memory and the accumulator.
System stores the result in the accumulator.
Divide or Multiply Accumulator by Value in Memory

Actor: User
System Goal: Perform division or multiplication between accumulator and memory
Preconditions: User wants to divide or multiply the value in the accumulator by a value in memory

System retrieves the value from memory at the specified address.
System performs the division or multiplication operation between the value in memory and the accumulator.
System stores the result in the accumulator.
Handle Large Arithmetic Results

Actor: System
System Goal: Handle large arithmetic results
Preconditions: The result of an arithmetic operation exceeds four digits
System handles the large result appropriately, such as by truncating, rounding, or displaying an error message.
Branch to a Distant Location

Actor: User
System Goal: Perform a branch to a distant location
Preconditions: User wants to branch to a location far away from the instruction pointer
System performs the branch instruction to the specified location.
Branch to a Distant Location if Accumulator is Negative

Actor: User
System Goal: Perform a branch to a distant location if the accumulator is negative
Preconditions: User wants to branch to a location far away from the instruction pointer if the accumulator is negative
System checks if the accumulator is negative.
If the accumulator is negative, system performs the branch instruction to the specified location.
Branch to a Distant Location if Accumulator is Zero

Actor: User
System Goal: Perform a branch to a distant location if the accumulator is zero
Preconditions: User wants to branch to a location far away from the instruction pointer if the accumulator is zero
System checks if the accumulator is zero.
If the accumulator is zero, system performs the branch instruction to the specified location.
Halt the Program

Actor: User
System Goal: Halt the program
Preconditions: User wants to stop the execution of the program
System terminates the program execution.