Alya Mohammed Al-Habibi 20230800

Jana Hassan Ali 20230106

Salah Hammad Salah Mohammed 20230194

**Design Document for Binary Calculator Project**

Project Overview

The Binary Calculator is designed to perform operations on binary numbers, including one's complement, two's complement, addition, and subtraction, without converting them to decimal. It offers a menu-driven user interface to interactively perform these operations.

Algorithms and Functions

Validation of Binary Numbers

Function: is\_valid\_binary(number)

Algorithm: Iterates through each character in the input string, checking if it is either '0' or '1'. Returns False if any other character is found, otherwise True.

Zero-Filling to Equal Length

Function: fill\_zero(bin1, length)

Algorithm: Prepends zeros to the binary string (bin1) until it reaches the specified length.

One's Complement

Function: ones\_complement(bin1)

Algorithm: Iterates through each bit of the binary number, flipping '0' to '1' and vice versa.

Two's Complement

Function: twos\_complement(bin1)

Algorithm: Calculates the one's complement of the binary number using ones\_complement(bin1) and then adds '1' to the result using the add\_binary function.

Binary Addition

Function: add\_binary(bin1, bin2)

Algorithm:

Equalize the lengths of both binary numbers.

Iteratively add corresponding bits of both numbers from right to left.

Maintain a carry which is added in each step.

Append the sum bit (or carry) to the result.

If there is a carry left at the end, append it to the result.

Binary Subtraction

Function: subtract\_binary(bin1, bin2)

Algorithm: Subtracts two binary numbers by adding the first number with the two's complement of the second. Trims any overflow bits to match the length of the original numbers.

Main Menu and Operations

Function: main\_menu()

Algorithm:

Displays a menu for the user to choose operations or exit.

Validates user choices and loops back for invalid inputs.

Calls respective functions based on user choices for different operations.

User Interaction Flow

The program displays the main menu with options to insert numbers or exit.

If the user chooses to insert numbers, they are prompted to enter a binary number. The number is validated for its binary format.

The user then selects an operation (one's complement, two's complement, addition, subtraction).

For addition and subtraction, the user is prompted to enter a second binary number.

The program performs the selected operation and displays the result.

The program loops back to the main menu allowing further operations or exit.

**Pseudo code for the problems:**

**Problem 1**:

Define the function INPUT\_NUMBER\_TO\_DECIMAL(number, base):

Set decimal\_number to 0

Set number to input string number. convert the number to its corresponding uppercase letter.

Set digits as a string variable

Assign the value to “0123456789ABCDEF” to the variable digit

for digit := number do:

set decimal\_number to (decimal\_number × base + digits.find the index in the digits list then add index value to decimal number

Return decimal\_number

Define the function DECIMAL\_TO\_OUTPUT\_NUMBER(number, base):

Set digits as a string variable

Assign the value to “0123456789ABCDEF” to the variable digit

If number is equal to 0:

Return “0 “

End if

Set converted\_number as an empty string’ ‘

While number is greater than zero:

Output set converted\_number = number MOD base then find digits of them + converted\_number

Set number DIV base

Return converted\_number

End while

Define the function MAIN\_MENU():

While true then

Print “\*\*Numbering system converter\*\*”;

Print “A) insert a new number”;

Print “B) Exit program”;

Set choice to input string “Enter your choice”. Remove leading white space from input then convert input to uppercase

End while

If choice is equal to ‘A’ then:

Set number to input “please insert a number: “ then remove leading white space from input

Set form\_base to base\_menu “form”;

Set decimal\_number to INPUT\_NUMBER\_TO\_DECIMAL(number, form base)

Set converted\_number to DECIMAL\_TO\_OUTPUT\_NUMBER(decimal\_number, to\_base)

Print “converted\_number: “ followed by the value of converted number;

Else if choice is equal to ‘B’ then:

Print “Exiting the program”;

Exit the loop

Else:

Print “please select a valid choice”;

End if

Define the function BASE\_MENU(direction):

While true then

Print “\*\*Please select the base you want to convert number followed by direction\*\*”

Print “A) decimal”;

Print “b) binary “;

Print “C) octal “;

Print “D) hexadecimal”;

Set choice to input string “Enter your choice” then remove leading white space from input and convert input to uppercase

End while

Create a dictionary named based\_dict:

* ‘A’ maps to 10
* ‘B’ maps to 2
* ‘C’ maps to 8
* ‘D’ maps to 16

If choice is := base\_dict:

Return the value corresponding to the choice from base\_dict

Else:

Print “please select a valid choice”;

End if

#start the main menu

Main\_Menu()

**Problem 2:**

Define the function FILL\_ZERO(bin1, length):

Return 0 × (length – bin1 until it reaches the specified length)

Define the function IS\_VALID\_BINARY(number):

For character := number:

If character not in ‘0’ or ‘1’:

Return False

Return True

End if

Define the function MAXIMUM(n1, n2):

Set m to n1

If n2 is greater than 0 then

m is equal to n2

Return m

End if

Define the function ADD\_BINARY(bin1, bin2):

Set max\_len to maximum of lengths of bin1 and bin2

Match fill\_zero with bin1 and max\_len then set bin1 to the result

Match fill\_zero with bin2 and max\_len then set bin2 to the result

set result to an empty string

set carry to 0

for i := max\_len-1 to -1

set r to carry

end for

if the character bin1 at index i is equal to ‘1’ then

r by 1

if the character bin2 at index i is equal to ‘1’ then

r by 1

if r MOD 2 is equal to 1 then

set result to ‘1’ + result

else

Set result to ‘0’ + result

End if

If r is less than 2 then

Set carry to 1

If carry is not equal to 0 then

Set result to ‘1’ + result

Return the output of fill\_zero with result and max\_len

Define the function SUBTRACTION\_BINARY(bin1, bin2) then

Match ADD\_BINARY bin1 and two’s complement of bin2 to the result

If the length of result is greater than the length of bin1 then

Remove the first character from the result

Return result

End if

Define the function ONE’S COMLEMENT(bin1):

Set result to an empty string

For x := bin1 then

If x is equal to ‘0’ then

Result by ‘1’

Else

Result by ‘0’

Return result

End if

End for

Define the function TWO’S COMPLEMENT(bin1) then

Return the result of adding ones complement and bin1 to ‘1’

Define the function MAIN\_MENU():

While true then

Print “a new line followed by 30 equals signs”;

Print “ \*\*Binary calculator \*\*”;

Print “ 30 equal signs ”;

Print “ a new line followed by 30 hyphens”;

Print “ A) Insert new numbers”;

Print “ B) Exit”;

Print “30 hyphens”

Set choice to input” Enter your choice: “

If choice is equal to ‘A’ then

Set bin1 to input “Enter the first binary number: “

End if

While not IS\_VALID\_BINARY(bin1) then

Set bin1 to input” please insert a valid binary number: “

End while not

While true then

Print “a new line followed by 30 equals signs”;

Print “ \*\* operations \*\*”

Print “ 30 equal signs ”;

Print “ a new line followed by 30 hyphens”;

Print “A) compute one’s complement”

Print “B) compute two’s complement”

Print “C) Addition”

Print “D) subtraction”

Print “30 hyphens”

Set operation to input “ Enter your choice: “ then convert input to uppercase

If operation is ‘A’, ‘B’, ‘C’ or ‘D’ then

Exit the loop

Else then

Print “operation not valid”

End if

If operation is equal to ‘A’ then

Print “One’s complement followed by the ones\_complement of bin1”;

Else if operation is equal to ‘B’ then

Print “Two’s complement followed by twos\_complement of bin1”;

Else if operation := ‘C’ or ‘D’ then

Set bin2 to input “please insert the second binary number: ”

End if

While not IS\_VALID\_BINARY(bin2) then

Set bin2 to input “please insert a valid binary number: “

If operation is equal to ‘C’ then

Print “Addition result followed by add\_binary bin1 and bin2”;

Else if operation is equal to ‘D’ then

Print” Subtraction result followed be subtract\_binary bin1 and bin2”;

End if

Else if choice is equal to ‘B’ then

Print ” - - - - Exiting the program - - - -“

Exit the loop

Else

Print “ please select a valid choice. “

#start the main menu

MAIN\_MENU()