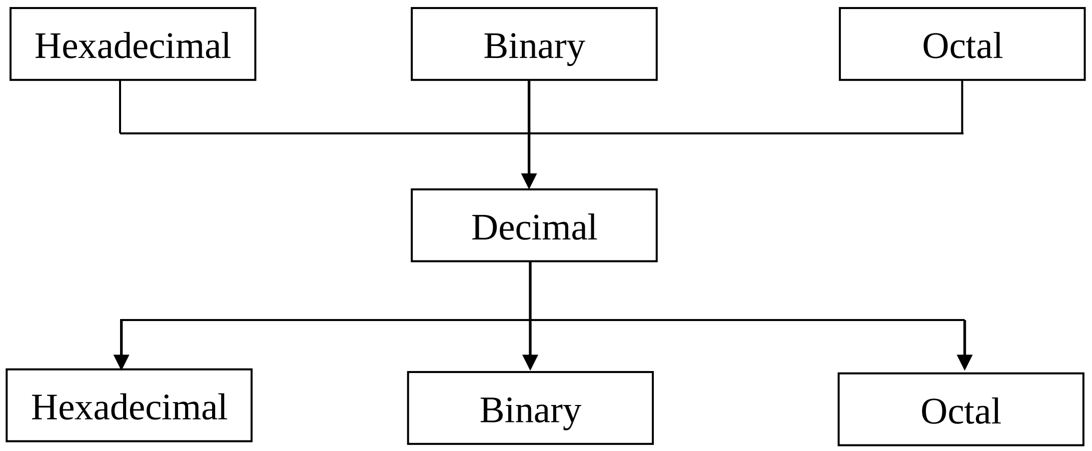
- Malak Amr 20230416
- Kirolos Adel 20230295
- Ahmed Tamer 20230012

Design Document

Problem 1: (Numbering System Convertor)

→ Steps:

- Show menu1 to make the user insert a number or exit the program.
- Check whether the user inserted a number or not using (is_a_number) function.
- If the user inserted a number, then menu2 appears to allow the user to choose the base he wants to convert from (dec, bin, oct, hex).
- Check the input value from menu2 to assign it to the right value using (main_bases) function and check whether the user inserted a valid choice or not, if not returns him to menu2 to insert a valid choice.
- Check whether the user inserted the right base or not by (is_valid) function.
- Assign the value of the base the user chose to the base variable as followes:
    - Choice (A) Decimal => assign base = 10
    - Choice (B) Binary => assign base =2
    - Choice (C) Octal => assign base = 8
    - Choice (D) Hexadecimal => assign base = 16
- Show menu3 to allow the user to choose the base he wants to convert to and checks if he inserted a valid choice or not by (main_bases) function, if not, it returns him to menu3 again to choose a valid choice.
- Finally, convert any number the user inserted to decimal and then from decimal to any system the user wants and prints final answer.



- Then return user to menu1 to choose whether he wants to do another conversion or exit.

→ Functions:

1. Converting from decimal to any base functions:

# Converting from decimal to binary function

```
Function Dec_to_bin (integer number):
Assign bases = “01”
Assign result = “ ”
    while (number greater than 0):
        Assign result = bases[number %2] + result
        Assign number = number /2
     End while
Return result value
```

# Converting from decimal to octal function

```
Function Dec_to_oct (integer number):
Assign bases = “0123456789”
Assign result = “ ”
    while (number greater than 0):
        Assign result = bases[number %8] + result
        Assign number = number /8
    End while
Return result value
```

# Converting from decimal to hexadecimal function

```
Function Dec_to_Hex (integer number):
Assign bases = “0123456789ABCDEF”
Assign result = “ ”
    while (number greater than 0):
        Assign result = bases[number %16] + result
        Assign number = number /16
    End while
Return result value
```

2. Converting from any base to decimal functions:

# Converting from binary to decimal function

```
Function Bin_to_Dec (integer number):
Assign i = 0
Assign result = 0
    while (number greater than 0):
        Assign last_digit = number % 10
        Assign result += last_digit * power(2^i)
        Assign number = number /10
        Increment i
    End while
Return result value
```

# Converting from octal to decimal function

```
Function Oct_to_Dec (integer number):
Assign i = 0
Assign result = 0
    while (number greater than 0):
        Assign last_digit = number % 10
        Assign result += last_digit * power(8^i)
        Assign number = number / 10
        Increment i
    End while
Return result value
```

# Converting from hexadecimal to decimal function

```
Function Hexa_to_Dec(integer number):
Assign hexa = “0123456789ABCDEF”
Assign result = 0
    For digit in hex_num:
        Assign  result  = result * 16 + hexa.index(digit )
    End for
Return result value
```

3. Validation Functions:

# Checking if the base of the number the user inserted is valid or not function

```
Function is_valid(integer user_num, integer base)
Assign  bases = “0123456789ABCDEF”
Assign  bases = bases[:base]
#Take the base form user then start from 0 to the number of base and store new value in bases
   For assign character in user_num:
        If character not in bases then
        Return false
  End for
Return true
```

# Checking if the user chose a valid choice from bases menu or not and then assign the value of the base the user chose function

```
Function main_bases(string msg, string menu)
While True loop:
       Assign bases = input (msg + menu).upper() # to make any letter from the user capitalized
       If base = ‘A’ then
                Base = 10
       Else if base = ‘B’ then
                Base = 2
       Else if base = ‘C’ then
                Base = 8
       Else if base = ‘D’ then
                Base = 16
       Else
                Show error message(“Error: please select a valid choice”)
                Then repeat the loop again
End while loop
Return base
```

# Checking whether the user inserted a number or not function

```
Function is_a_number (integer user_number)
Assign bases = “0123456789ABCDEF”
If not user_number then
    Return false
For assign character in user_number
       If character not in bases then
       Return false
End for
Return true
```

→ Main Program

```
First while True loop:
    # Menu1 = A) Insert a new number
                   B) Exit program
    Show menu1 to ask user if he wants to choose (A) or (B)
    Take all input from user capitalized by [.upper()] to avoid errors if user input small case letters
        If user chose A => Ask the user to insert a number
        If user chose B => exit the program
        If the user does not choose A or B => show an error message
        ("Error! Please select a valid choice")


Second while True loop:
    Ask the user to insert a number and store it in (main_number) variable
        If the user doesn't insert any value, show an error message
          ("Error! Please insert a valid number")
        Then repeat the while loop again
        If user inserted a number => end the loop and show menu2
    # Menu2 = "Please select the base you want to convert a number from"
                     A) Decimal
                     B) Binary
                     C) Octal
                     D) Hexadecimal
        If user choose A => assign base = 10
        If user choose B => assign base = 2
        If user choose C => assign base = 8
        If user choose D => assign base = 16
        If the user does not choose any value from above show an error message
         ("Error! Please select a valid choice")


 Third while loop:
    Assign first_base = base (10,2,8,16)
    Check if it is a valid base for the number the user inserted
        If the number is not valid show error message
        ("Error in base number, << show the base >> please insert the right base for the
        number you inserted")
    And repeat the while loop again
        If the number is valid => end the loop and show menu3
    # Menu3 = "Please select the base you want to convert a number to"
                     A) Decimal
                     B) Binary
                     C) Octal
                     D) Hexadecimal
```

```
Assign second_base = base  # that base come from menu three

# Convert the number the user inserted to decimal
        If first_base = 10
              Assign user_number = integer(user_number)
        If first_base = 2
              Assign user_number = call function bin_to_dec (user_number)
        If first_base = 8
              Assign user_number = call function oct_to_dec(user_number)
        If first_base = 16
              Assign user_number = call function hex_to_dec (user_number)



# Convert the number from decimal to the base the user chose
        If second_base = 2
              Assign user_number = call function dec_to_bin (user_number)
        If second_base = 8
              Assign user_number = call function dec_to_oct (user_number)
        If second_base = 16
              Assign user_number = call function dec_to_hex(user_number)

Show the final result ( "The result is: << user_number>>)
Then Show menu1 again to let the user make another conversion or exit the program
```

Problem 2: (Binary Calculator)

→ Steps:

- Show menu1 to make the user insert a number or exit the program
- Check whether the number the user inserted is a valid binary number or not using (is_valid) function
- If the number is valid show menu2 to let the user choose the operation he wants
- If the number is not valid ask the user again to insert a valid binary number
- If the user chose any of the complements => show the result and then show menu1 again
- If the user chose addition or subtraction => ask him to insert the second number
- Check whether the number the user inserted is a valid binary number or not using (is_valid) function
- If the number is valid show the result and then return to menu1
- If the number is not valid ask the user to insert a valid binary number.
- Return to men1 until the user choose to exit the program

→ Functions:

1. Validation function

```
# Check if the user inserted a valid binary number or not
   Function is_valid (number)
   Assign binary_digits = "01"
   If all digits in number are in binary_digits
      Return True
   Else
Return False
```

2.Operations functions:

```
    # First complement function

  Function complement_one (integer num)
  Assign result = ""
      For char in num:
          if char equal "0"
           Assign result = result + "1"
       else
            Assign result = result + "0"
      End for
  Return result value
```

```
 # Second complement function

   Function complement_two (num):
   Assign result = " "
   Assign flip = False
   For i in reversed (num):
     if (flip)
         if i == "0":
            Assign i = "1"
         else
            Assign i = "0"
     if i == "1":
         Assign flip = True
     Assign result = i + result
   End for
   Return result value
```

```
# Addition function

    Function addition(string_ bin1, string_bin2):
    Assign width = maximum number of characters between bin1 and bin2
    Assign bin1 = fill the empty characters with 0
    Assign bin2 = fill the empty characters with 0
    Assign result = ""
    Assign carry = 0
         For i in reversed width starts from 0
             Assign res = carry
             if bin1[i] equal '1'
                   Assign res = rest + 1
             else
                   Assign res = res + 0
             if bin2[i] equal '1'
                   Increment res
             else
                   Assign res = res + 0
             if res % 2 equal 1
                   Assign result = '1' + result
             else
                   Assign result = '0' + result
             if res < 2
                   Assign carry = 0
             else
                   Assign carry = 1
         if carry != 0:
         Assign result = '1' + result
         End for
Return result value
```

```
# Subtraction function

  Function subtraction(string binary_num1,string binary_num2):
  Assign max_length = maximum number of characters between bin1 and bin2
  Assign binary_num1 = fill the empty characters with 0
  Assign binary_num2 = fill the empty characters with 0
  Assign borrow = 0
  Assign result = “ ”
    For i to reversed max_length starts from 0
      Assign num1 = binary_num1[i] as integer
      Assign num2 = binary_num2[i] as integer
      Set difference = num1 - num2 - borrow
      Set final_answer = (difference + 2) % 2) as string value   # To avoid negative numbers
    if difference < 0
      Assign borrow = 1
    else:
      Assign borrow = 0
    Assign result = final_answer + result
    End for
  Return result value
```

→ Main Program

```
First while True loop:
# Menu1 = A) Inset new number
          B) Exit program
Show menu1 to ask user whether he wants to insert a number or exit
If user didn't choose A or B, show an error message
(Error! Please insert a valid choice)
If user chose (A) => end loop and then ask the user to insert a number


Second while True loop:
Take the number user inserted and check if it is a valid binary number or not
If it is not valid, show error message
(Error! Please insert a valid binary number) and ask him to insert a number again
If the user didn't insert a number, show error message
(Error! Please insert a number) and ask him to insert a number again
If it is a valid number => store the number in main_number end loop and then show
menu2

Third while True loop:
# Menu2 = "Please select an option"
          A)  First complement
          B)  second complement
          C)  Addition
          D)  Subtraction
 Take the choice the user inserted and store it in choice_menu2
 If choice_menu2 == "A"
     Assign answer = call first_complement (main_number)
     Show Final anwer ("The result = ", << answer >>)
     Return to menu1 to let the user choose if he wants to do another operation or exit

 If choice_menu2 == "B"
     Assign answer = call second_complement (main_number)
     Show Final anwer ("The result = ", << answer >>)
     Return to menu1 to let the user choose if he wants to do another operation or exit
```

```
If choice_menu2 = "C"
    # Define another while true loop to take the second number and check its validation
      while True:
       Ask the user to insert the second number and store it in second_number variable
       If the user didn't insert a number, show error message
       (Error! Please insert a number)
       If the number is not valid, show error message
       (Error! Please insert a valid binary number) and ask him to insert a number again
       If the number is valid => end loop then
       Assign answer = call addition (main_number, second_number)
       Show Final anwer ("The result = ", << answer >>)
       Return to menu1 to let the user choose if he wants to do another operation or exit


If choice_menu2 = "D"
    # Define another while true loop to take the second number and check its validation
      while True:
       Ask the user to insert the second number and store it in second_number variable
       If the user didn't insert a number, show error message
       (Error! Please insert a number)
       If the number is not valid, show error message
       (Error! Please insert a valid binary number) and ask him to insert a number again
       If the number is valid => end loop then
       Assign answer = call subtraction (main_number, second_number)
       Show Final anwer ("The result = ", << answer >>)
       Return to menu1 to let the user choose if he wants to do another operation or exit

If the user didn't insert a choice in menu2, show error message
(Error! Please insert a valid choice) and return the user to menu2 to choose a valid choice
```