# UITS

## UNIVERSITY OF INFORMATION TECHNOLOGY AND SCIENCES

# Number System Conversion Project Report

**Submitted To :**

**Name** : Saima Siddique Tashfia

**Designation** : Lecturer of uits

**Date of Submission :** 09/12/2024

**Cours code** : CSE 412

**Course title :** Operating Systems Lab

**Submitted By :**

**Name :** Md Shariful islam sajib sarker (2125051016)

**Name :** Kamrun nahar mitu (2125051019)

**Name :** Kamrul Huda Sattari Saad (2125051041)

**Name :** Md Tajbeer Ahamed Rimon (2125051007)

**Batch** : 50

# Table Of Contents

# 1. Introduction

Number system conversions are a critical aspect of computer science and software development, enabling programmers to work with various numerical representations effectively. This project demonstrates the implementation of a versatile Bash script designed to perform these conversions interactively.

The script features a menu-driven interface where users can select the type of conversion they wish to perform, enter the number, and receive the result instantly. It supports conversions between Binary, Decimal, Octal, and Hexadecimal number systems, and includes a custom option for conversions between any of these bases. The user can also exit the program at any point, ensuring a seamless experience.

By leveraging the capabilities of Bash scripting and built-in Linux tools, this project highlights how command-line scripts can be used to solve practical computational problems efficiently.

# 2. Background

In the realm of computing, number systems such as Binary (Base-2), Decimal (Base-10), Octal (Base-8), and Hexadecimal (Base-16) play a fundamental role in data representation and computation. Conversions between these systems are often required in various scenarios, including:

- Debugging low-level programs.

- Understanding memory addresses.

- Interpreting machine code.

Traditional number system conversion involves manual calculations or the use of specialized tools. However, Bash scripting provides a lightweight, scriptable alternative for performing such conversions directly in the command line, offering the benefits of flexibility, automation, and portability.

This project seeks to implement a robust script that supports these conversions, incorporating validation and error handling to ensure accuracy and reliability.

# 3. Methodology

The Bash script was designed and implemented with the following key steps:

## 3.1 Menu Design

A user-friendly menu was created to display the available conversion options. The menu includes:

- **Decimal to Binary**

- **Decimal to Octal**

- **Decimal to Hexadecimal**

- **Binary to Decimal**

- **Octal to Decimal**

- **Hexadecimal to Decimal**

- **Custom Conversion (Any Base to Any Base)**

- **Exit**

The menu is displayed in a loop, allowing the user to perform multiple conversions without restarting the program.

## 3.2 Input Validation

A function was implemented to validate input numbers based on their base. For example:

- Binary numbers must contain only 0 and 1.

- Octal numbers must contain digits from 0 to 7.

- Hexadecimal numbers can contain digits 0-9 and letters A-F.

## 3.3 Conversion Logic

The script performs conversions in two steps:

1. Convert the input number to its Decimal equivalent.

2. Convert the Decimal number to the desired output base.

This approach ensures that any base-to-base conversion is possible using intermediate Decimal representation.

## 3.4 Error Handling

The script checks for invalid inputs and provides appropriate error messages, prompting the user to try again. This ensures a smooth user experience and prevents incorrect results.

## 3.5 Custom Base-to-Base Conversion

The custom conversion option allows users to specify any input and output base (from Binary, Octal, Decimal, or Hexadecimal) and convert between them. This flexibility makes the script highly versatile.

## 3.6 Looping Mechanism

The main program loop ensures that the menu is displayed repeatedly until the user selects the "Exit" option, enabling continuous usage without restarting the program.

# 4. Results/Output

The script was tested extensively with various inputs, and the results were accurate for all supported conversions. Below is the complete code:

```bash
#!/bin/bash

display_menu() {

    echo "Number System Conversion Tool"

    echo "1. Decimal to Binary"
```

```bash
        echo "2. Decimal to Octal"

        echo "3. Decimal to Hexadecimal"

        echo "4. Binary to Decimal"

        echo "5. Octal to Decimal"

        echo "6. Hexadecimal to Decimal"

        echo "7. Custom Conversion (Any to Any)"

        echo "8. Exit"

        echo -n "Choose an option: "
}
is_valid_number() {

        local number=$1

        local base=$2

        case $base in

                2)  [[ $number =~ ^[01]+$ ]] ;;

                8)  [[ $number =~ ^[0-7]+$ ]] ;;

                10) [[ $number =~ ^[0-9]+$ ]] ;;

                16) [[ $number =~ ^[0-9A-Fa-f]+$ ]] ;;

                *)  return 1 ;; # Invalid base

        esac
}
convert_from_decimal() {

        local decimal=$1

        local base=$2

        local result=""

        local digits="0123456789ABCDEF"
```

```bash
    while [ "$decimal" -gt 0 ]; do

        remainder=$((decimal % base))

        result="${digits:remainder:1}$result"

        decimal=$((decimal / base))

    done

    echo "${result:-0}" # Return 0 if result is empty

}

convert_to_decimal() {

    local number=$1

    local base=$2

    local length=${#number}

    local decimal=0

    local digits="0123456789ABCDEF"

    for (( i=0; i<length; i++ )); do

        char=${number:i:1}

        value=$(expr index "$digits" "${char^^}") # Convert to uppercase for
Hex

        value=$((value - 1))  # `expr index` is 1-based

        decimal=$((decimal * base + value))

    done

    echo "$decimal"

}

convert_base_to_base() {

    local input_base=$1

    local output_base=$2

    local number=$3
```

```
    if ! is_valid_number "$number" "$input_base"; then

        echo "Invalid number '$number' for base $input_base. Please try
again."

        return 1

    fi

    decimal=$(convert_to_decimal "$number" "$input_base")

    result=$(convert_from_decimal "$decimal" "$output_base")

    echo "$result"

}

process_conversion() {

    case $1 in

        1)

            echo -n "Enter a Decimal number: "

            read decimal

            if is_valid_number "$decimal" 10; then

                echo "Binary: $(convert_from_decimal "$decimal" 2)"

            else

                echo "Invalid Decimal number. Please try again."

            fi

            ;;

        2)

            echo -n "Enter a Decimal number: "

            read decimal

            if is_valid_number "$decimal" 10; then

                echo "Octal: $(convert_from_decimal "$decimal" 8)"

            else
```

```
                        echo "Invalid Decimal number. Please try again."

                fi

                ;;

        3)

                echo -n "Enter a Decimal number: "

                read decimal

                if is_valid_number "$decimal" 10; then

                        echo "Hexadecimal: $(convert_from_decimal "$decimal" 16)"

                else

                        echo "Invalid Decimal number. Please try again."

                fi

                ;;

        4)

                echo -n "Enter a Binary number: "

                read binary

                if is_valid_number "$binary" 2; then

                        echo "Decimal: $(convert_to_decimal "$binary" 2)"

                else

                        echo "Invalid Binary number. Please try again."

                fi

                ;;

        5)

                echo -n "Enter an Octal number: "

                read octal

                if is_valid_number "$octal" 8; then

                        echo "Decimal: $(convert_to_decimal "$octal" 8)"
```

```
        else

            echo "Invalid Octal number. Please try again."

        fi

        ;;

    6)

        echo -n "Enter a Hexadecimal number: "

        read hex

        if is_valid_number "$hex" 16; then

            echo "Decimal: $(convert_to_decimal "$hex" 16)"

        else

            echo "Invalid Hexadecimal number. Please try again."

        fi

        ;;

    7)

        echo -n "Enter the Input Base (2, 8, 10, or 16): "

        read input_base

        echo -n "Enter the Output Base (2, 8, 10, or 16): "

        read output_base

        echo -n "Enter the Number to Convert: "

        read number

        if ! [[ "$input_base" =~ ^(2|8|10|16)$ ]] || ! [[ "$output_base"
=~ ^(2|8|10|16)$ ]]; then

            echo "Invalid base(s). Please enter 2, 8, 10, or 16."

        else

            result=$(convert_base_to_base  "$input_base"  "$output_base"
"$number")
```

```bash
                if [ $? -ne 0 ]; then

                    echo "Conversion failed due to invalid input. Please try
again."

                else

                    echo "Converted Number: $result"

                fi

            fi

            ;;

        8)

            echo "Exiting..."; exit 0

            ;;

        *)

            echo "Invalid option. Please try again."

            ;;

    esac
}
while true; do

    display_menu

    read choice

    process_conversion $choice

    echo ""

done
```
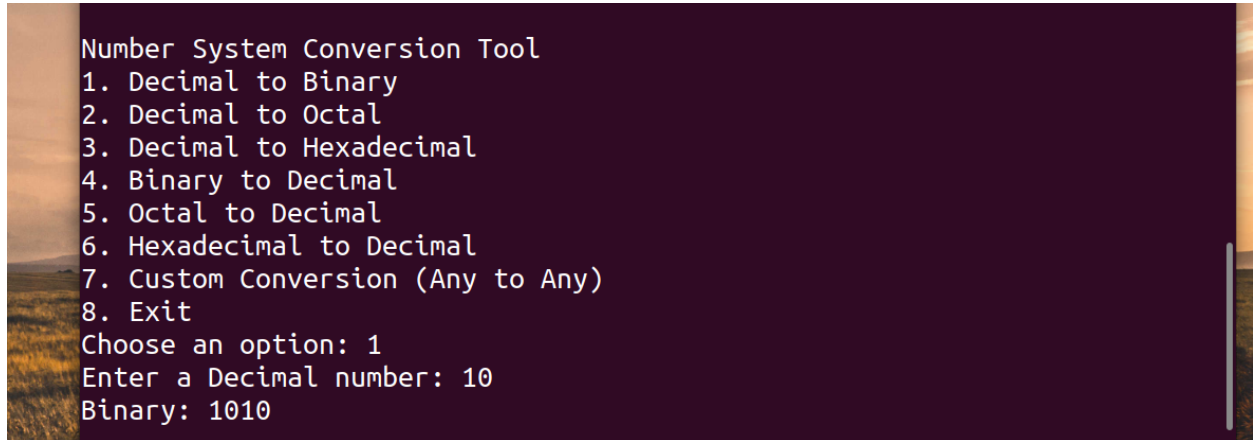
## 4.1 Sample Runs

**Case 1: Decimal to Binary**

```
Number System Conversion Tool
1. Decimal to Binary
2. Decimal to Octal
3. Decimal to Hexadecimal
4. Binary to Decimal
5. Octal to Decimal
6. Hexadecimal to Decimal
7. Custom Conversion (Any to Any)
8. Exit
Choose an option: 1
Enter a Decimal number: 10
Binary: 1010
```
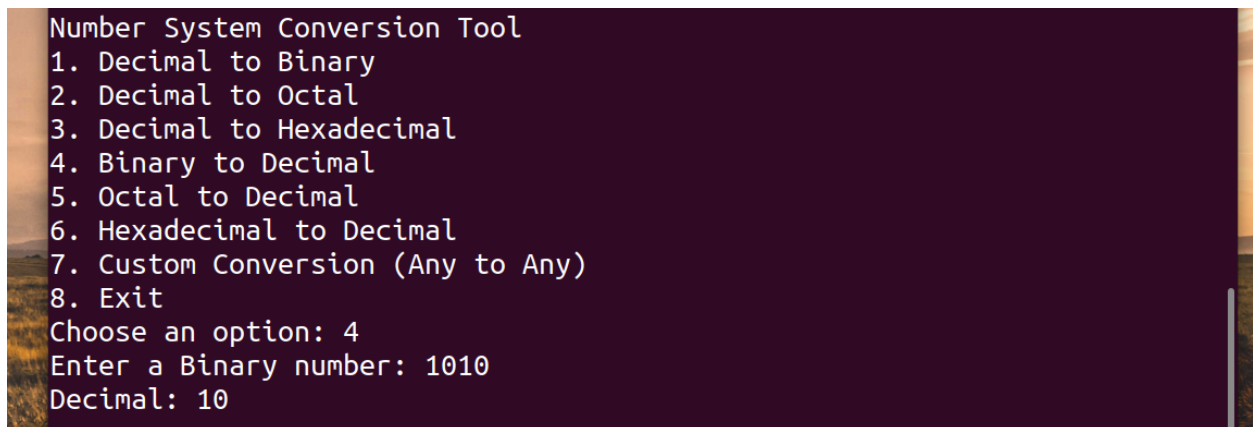
**Input:**

- Decimal Number: 10

**Output:**

- Binary: 1010

**Case 2: Binary to Decimal**

```
Number System Conversion Tool
1. Decimal to Binary
2. Decimal to Octal
3. Decimal to Hexadecimal
4. Binary to Decimal
5. Octal to Decimal
6. Hexadecimal to Decimal
7. Custom Conversion (Any to Any)
8. Exit
Choose an option: 4
Enter a Binary number: 1010
Decimal: 10
```
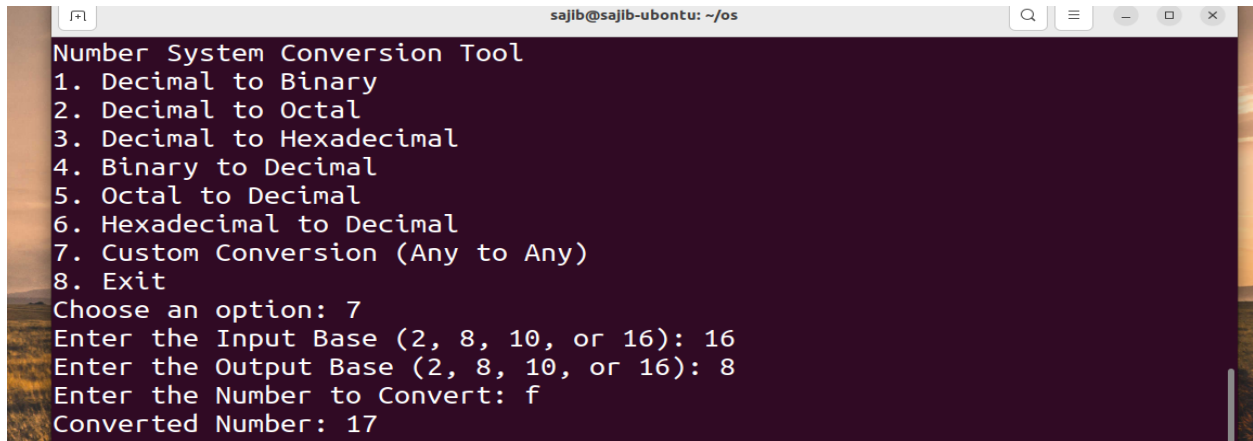
**Input:**

- Binary Number: 1010

**Output:**

- Decimal: 10

## Case 3: Custom Conversion



```
sajib@sajib-ubontu: ~/os
Number System Conversion Tool
1. Decimal to Binary
2. Decimal to Octal
3. Decimal to Hexadecimal
4. Binary to Decimal
5. Octal to Decimal
6. Hexadecimal to Decimal
7. Custom Conversion (Any to Any)
8. Exit
Choose an option: 7
Enter the Input Base (2, 8, 10, or 16): 16
Enter the Output Base (2, 8, 10, or 16): 8
Enter the Number to Convert: f
Converted Number: 17
```
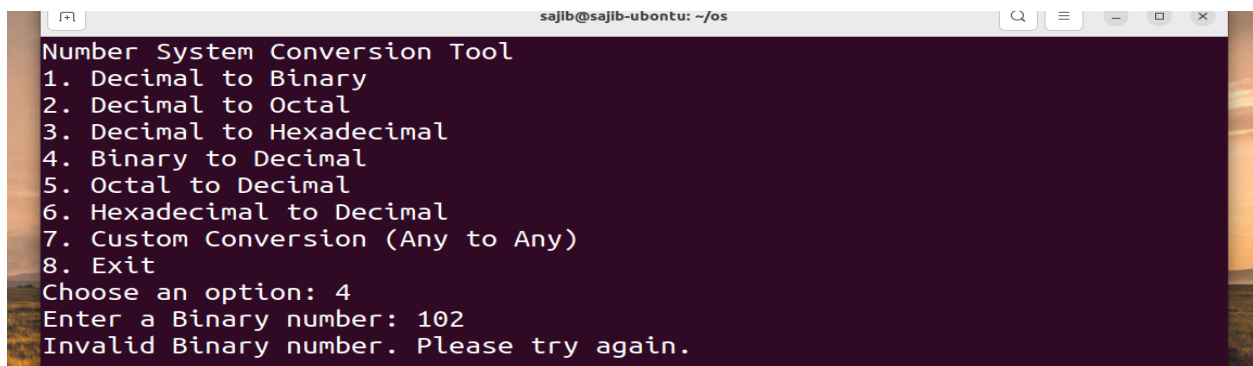
**Input:**

- Input Base: 16 (Hexadecimal)

- Output Base: 8 (Octal)

- Number: F

**Output:**

- Converted Number: 17

## Case 4: Invalid Input



```
sajib@sajib-ubontu: ~/os
Number System Conversion Tool
1. Decimal to Binary
2. Decimal to Octal
3. Decimal to Hexadecimal
4. Binary to Decimal
5. Octal to Decimal
6. Hexadecimal to Decimal
7. Custom Conversion (Any to Any)
8. Exit
Choose an option: 4
Enter a Binary number: 102
Invalid Binary number. Please try again.
```

**Input:**

- Binary Number: 102 (Invalid)

**Output:**

- Error: "Invalid Binary number. Please try again."

# 5. Conclusion and Future Work
## 5.1 Conclusion

This project successfully implemented a Bash script for performing number system conversions, offering a user-friendly interface and robust functionality. The script is efficient and reliable, capable of handling a wide range of conversions with proper validation and error handling.

Key learnings from this project include:

- Mastery of conditional statements (case) and looping constructs in Bash.

- Use of arithmetic and string manipulation techniques for numerical conversions.

- Implementation of input validation to ensure program correctness.

## 5.2 Future Work

To enhance the script further, the following improvements can be done:

1. **Batch Processing**: Allow the script to process multiple conversions from a file or input list.

2. **Enhanced UI**: Integrate color-coded outputs and animations to improve user experience.

3. **Extended Number Systems**: Support for additional bases, such as Base-3 or Base-7.

4. **Web Integration**: Develop a web interface for the script to make it accessible from browsers.

# 6. References

1. **GNU Bash Manual**: https://www.gnu.org/software/bash/manual/bash.html

2. **Linux Command Documentation**: https://linux.die.net/man/

3. Classroom Notes and Course Materials