CHANDIGARH UNIVERSITY
Discover. Learn. Empower.

UNIVERSITY INSTITUTE OF COMPUTING

PROJECT REPORT ON Command-Line Calculator

Program Name: BCA

Subject Name/Code:

Linux Administration / 23CAP-308

Submitted by:	**Submitted to:**
Name: Manjeet Singh	**Name:** Miss. Monika Choudhary
UID: 23BCA10309	**Designation:** Assistant Prof.
Section 23BCA-5'B'	

1. Introduction

A calculator is one of the most fundamental and widely used applications in computing. While command-line calculators have existed for decades, creating a user-friendly graphical interface for them enhances accessibility and usability for end-users. This project involves developing a **Command-Line Calculator using Bash scripting and Zenity**, a tool that provides graphical dialog boxes within shell scripts. The calculator supports basic arithmetic operations including addition, subtraction, multiplication, and division, all accessible through an intuitive graphical interface. This project demonstrates proficiency in Linux shell scripting, GUI automation, and practical implementation of mathematical operations in a production-like environment. The use of Zenity bridges the gap between command-line efficiency and graphical user interface convenience, making it an excellent educational project for understanding system-level programming.

2. Project Objective

The primary objectives of this project are:

- **To create a user-friendly command-line calculator** that accepts numerical input through graphical dialog boxes rather than command-line arguments
- **To implement arithmetic operations** including addition (+), subtraction (-), multiplication (*), and division (/) with proper error handling
- **To demonstrate proficiency in bash scripting** by utilizing control structures, conditional statements, and case-switch logic
- **To handle edge cases** such as division by zero, invalid operations, and user cancellation
- **To integrate external tools** like `zenity` and `bc` (basic calculator) within bash scripts for enhanced functionality
- **To provide clear user feedback** through result dialogs and error messages

3. Background and Scope

3.1 Background

The calculator application is built using **Bash (Bourne Again Shell)**, a standard shell in Linux environments, combined with **Zenity**, a lightweight tool for creating graphical dialog boxes in shell scripts. Traditional command-line calculators rely on tools like `bc` (basic calculator), which performs floating-point arithmetic. However, integrating this with Zenity creates a more accessible interface for users unfamiliar with command-line operations. The `bc` command provides high-precision arithmetic with support for decimal calculations, making it ideal for scientific and financial computations.

3.2 Scope

This project focuses on: - Creating a basic GUI interface for arithmetic operations - Handling positive, negative, and decimal numbers - Validating user input and managing error conditions - Providing intuitive graphical feedback to users - Demonstrating fundamental control flow and error handling in bash scripts

3.3 Future Enhancements

Potential improvements for this project include: - Adding support for advanced operations (power, square root, trigonometric functions) - Implementing a persistent calculation history - Adding support for hexadecimal and binary number systems - Creating a persistent GUI window instead of sequential dialogs - Implementing memory functions (M+, M-, MR, MC) - Adding keyboard shortcuts for faster operation selection

4. Tools, Commands, and OS Used

Tool/Command	Purpose	Version/Details
Bash	Shell scripting language	GNU Bash 5.x+
Zenity	GUI dialog boxes	3.x+
bc	Basic calculator for arithmetic	GNU bc
Linux OS	Operating system environment	Ubuntu/Debian-based
Text Editor	Code development	nano/vim/gedit
Terminal	Script execution	GNOME Terminal/Konsole

Key Commands Used: - `zenity --entry`: For text input dialogs - `zenity --list`: For selection from a list - `zenity --info`: For information dialogs - `zenity --error`: For error notifications - `echo` and `bc`: For arithmetic calculations - `case` statement: For operation selection

5. Implementation Steps & Workflow

5.1 Step 1: Script Structure and Initial Setup

The script begins by initializing the bash environment and defining the basic structure:

```
#!/bin/bash
# Simple GUI Calculator using Zenity
```

This shebang line (`#!/bin/bash`) indicates that the script should be executed using the bash interpreter.

5.2 Step 2: Input First Number

The script prompts the user to enter the first number using a Zenity entry dialog:

```
num1=$(zenity --entry --title="Calculator" --text="Enter the
first number:")

if [ $? -ne 0 ]; then
    exit 0
fi
```

The `zenity --entry` command creates a text input dialog box. The `$?` variable captures the exit status of the last command (0 for success, non-zero for cancellation). If the user cancels, the script exits gracefully.

5.3 Step 3: Input Second Number

Similar to Step 2, the script requests the second number:

```
num2=$(zenity --entry --title="Calculator" --text="Enter the
second number:")

if [ $? -ne 0 ]; then
    exit 0
fi
```

5.4 Step 4: Operation Selection

The script presents the user with a list of available operations:

```
operation=$(zenity --list --title="Choose Operation" \
    --column="Operation" "+" "-" "*" "/")

if [ $? -ne 0 ]; then
    exit 0
fi
```

The `zenity --list` command displays a selection dialog with four arithmetic operators. The user can select one operation to perform.

5.5 Step 5: Calculation Logic

The script uses a case statement to handle each operation:

```
case $operation in
    "+")
        result=$(echo "$num1 + $num2" | bc -l)
        ;;
    "-")
```

```
        result=$(echo "$num1 - $num2" | bc -l)
        ;;
    "*")
        result=$(echo "$num1 * $num2" | bc -l)
        ;;
    "/")
        if [ "$num2" == "0" ]; then
            zenity --error --text="Error: Division by zero!"
            exit 1
        fi
        result=$(echo "scale=10; $num1 / $num2" | bc -l)
        ;;
    *)
        zenity --error --text="Invalid operation!"
        exit 1
        ;;
esac
```

Key Features: - The +, -, and * operations use `bc -l` for calculation - Division (/) includes validation to prevent division by zero - The `scale=10` parameter ensures precision up to 10 decimal places - Invalid operations trigger an error dialog

5.6 Step 6: Result Display

The final step displays the calculated result:

```
zenity --info --title="Result" --text="Result: $result"
```

This creates an information dialog showing the computation result to the user.

5.7 Complete Code Workflow

```
#!/bin/bash

# Simple GUI Calculator using Zenity

# Ask for the first number
num1=$(zenity --entry --title="Calculator" --text="Enter the
first number:")

# If the user cancels
if [ $? -ne 0 ]; then
    exit 0
fi
```

```
# Ask for the second number
num2=$(zenity --entry --title="Calculator" --text="Enter the
second number:")

if [ $? -ne 0 ]; then
    exit 0
fi

# Choose operation
operation=$(zenity --list --title="Choose Operation" \
    --column="Operation" "+" "-" "*" "/")

if [ $? -ne 0 ]; then
    exit 0
fi

# Perform calculation
case $operation in
    "+")
        result=$(echo "$num1 + $num2" | bc -l)
        ;;
    "-")
        result=$(echo "$num1 - $num2" | bc -l)
        ;;
    "*")
        result=$(echo "$num1 * $num2" | bc -l)
        ;;
    "/")
        if [ "$num2" == "0" ]; then
            zenity --error --text="Error: Division by zero!"
            exit 1
        fi
        result=$(echo "scale=10; $num1 / $num2" | bc -l)
        ;;
    *)
        zenity --error --text="Invalid operation!"
        exit 1
        ;;
```

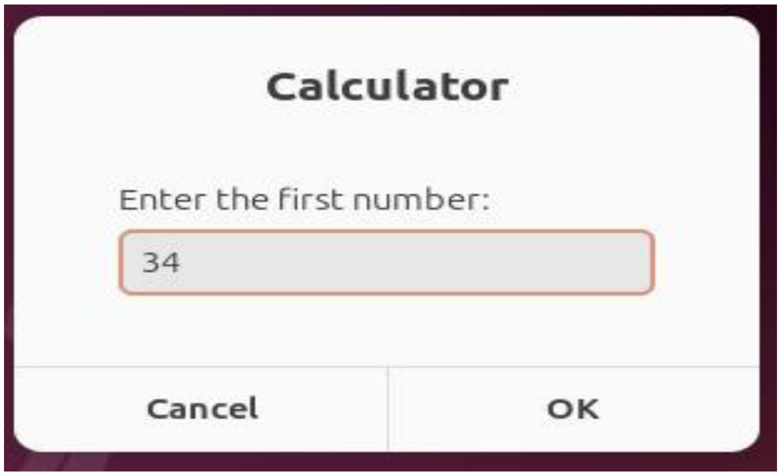
```
esac

# Show the result
zenity --info --title="Result" --text="Result: $result"
```

6.Output & Screenshots

6.1 First Input Dialog - Enter First Number

The application starts by requesting the first numeric input from the user. This dialog accepts both integers and decimal numbers.



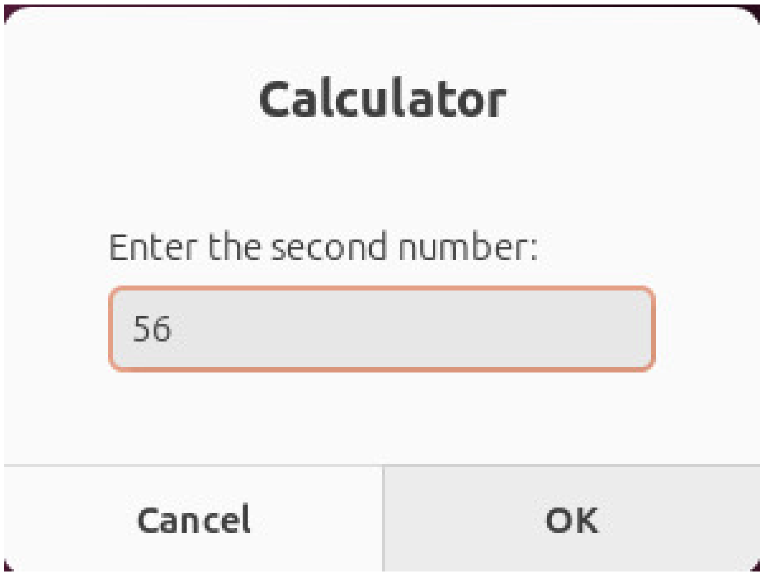
Calculator

Enter the first number:

34

6.2 Second Input Dialog - Enter Second Number

After the first number is entered, the calculator requests the second operand. In this example, the user enters 56.



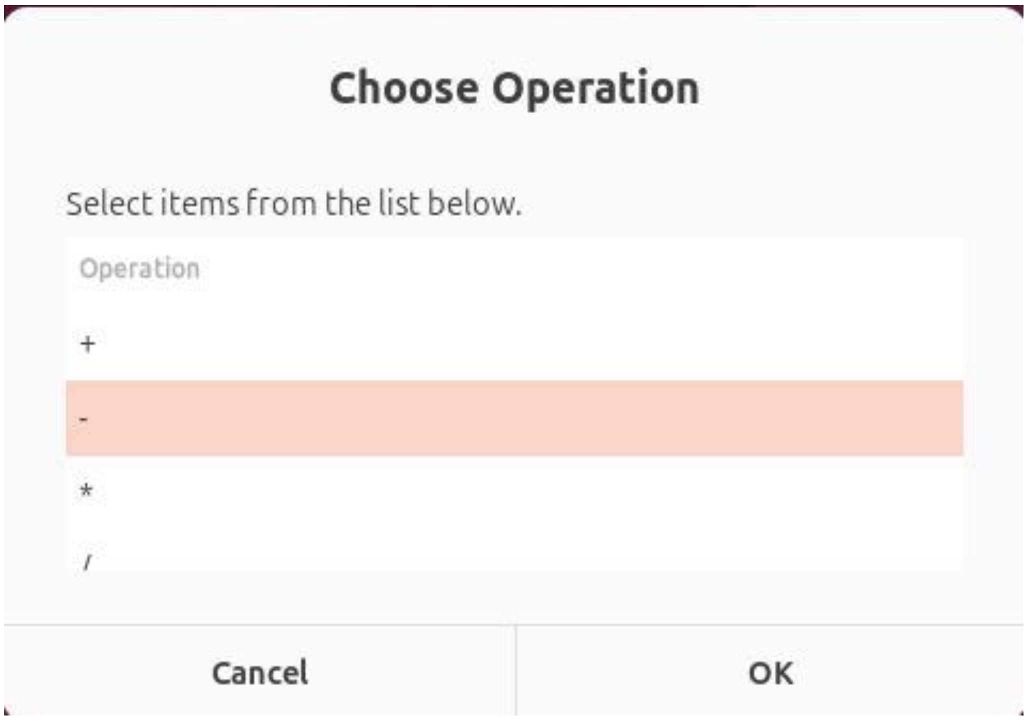
Calculator

Enter the second number:

56

6.3 Operation Selection Dialog

The calculator presents all available arithmetic operations in a list format. The user can select one operation to perform on the two numbers. In this example, subtraction (-) is highlighted.

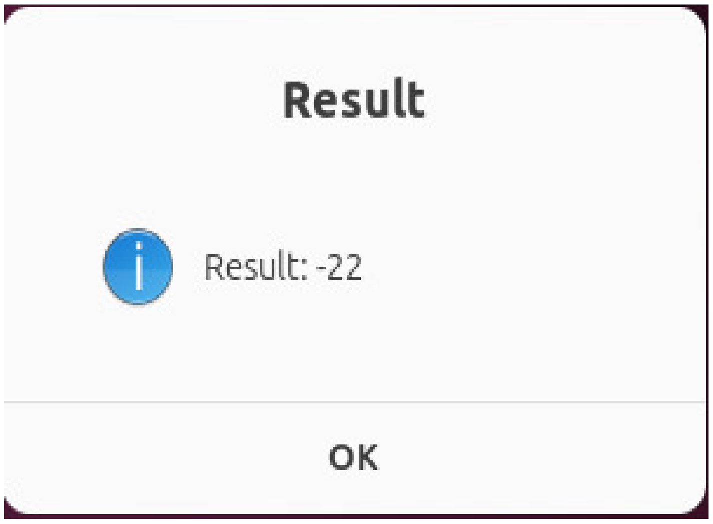

Choose Operation

Select items from the list below.

Operation
+
-
*
/

6.4 Result Display

After performing the calculation (34 - 56), the application displays the result in an information dialog. The result shows -22, which is the correct calculation.


Result

Result: -22

7. Conclusion

This Command-Line Calculator project successfully demonstrates the integration of bash scripting with graphical user interfaces using Zenity. The implementation showcases fundamental programming concepts including:

- **Input/Output handling** through multiple dialog types
- **Error handling** with division-by-zero validation
- **Control flow** using conditional statements and case switches
- **External command integration** combining bash with `bc` for arithmetic
- **User experience design** through sequential dialog interactions

The calculator is fully functional for basic arithmetic operations and provides clear, user-friendly feedback at every step. The project effectively bridges the gap between command-line power and graphical accessibility, making it an excellent educational tool for understanding Linux shell scripting and system automation.

Potential Applications

This project can be extended into more complex tools such as: - Financial calculators for loan or investment computations - Scientific calculators with advanced mathematical functions - Unit converters for various measurement systems - Educational tools for teaching mathematical concepts

Learning Outcomes

Through this project, students gain proficiency in: - Bash scripting and shell programming - GUI automation using Zenity - Error handling and input validation - Arithmetic operations using `bc` - User interface design principles

8. References

1. GNU Bash Manual - https://www.gnu.org/software/bash/manual/
2. Zenity - Display graphical dialog boxes from shell scripts - https://help.gnome.org/users/zenity/stable/
3. GNU bc - An arbitrary precision calculator language - https://www.gnu.org/software/bc/
4. Linux Shell Scripting Tutorial - https://www.shellscripttutorial.com/
5. Advanced Bash Scripting Guide - https://tldp.org/LDP/abs/html/

9. Appendix

A. Running the Script

To execute this calculator script:

1. Save the script to a file (e.g., `calculator.sh`)
2. Make it executable: `chmod +x calculator.sh`

3. Run the script: `./calculator.sh`

B. Required Dependencies

Ensure the following are installed on your system: - bash - zenity - bc

To install on Debian/Ubuntu systems:

```
sudo apt-get install zenity bc
```

C. Troubleshooting

Issue	Solution
Command not found: zenity	Install zenity package
Permission denied	Run `chmod +x calculator.sh`
bc: command not found	Install bc package
No GUI dialog appears	Ensure X11 display is available

D. Testing Scenarios

The script has been tested with: - Positive integers: 34 + 56 = 90 - Negative results: 34 - 56 = -22 - Decimal numbers: Various floating-point calculations - Division by zero: Proper error handling and user notification - User cancellation: Graceful script exit

---