PROJECT REPORT ON

"Simple Calculator"

Submitted By:

Pankaj Kumar Singh(UID- 24MCA20067)

Under The Guidance of:
Ms Prabhjot Kaur

October, 2024

University Institute of Computing

Chandigarh University,

Mohali, Punjab

CERTIFICATE

This is to certify that Pankaj Kumar Singh(UID- 24MCA20067) have successfully completed the project title “Simple Calculator in linux” at University Institute of Computing under my supervision and guidance in the fulfilment of requirements of first semester, Master of Computer Application-Specialization in General. Of Chandigarh University, Mohali, Punjab.

Dr. Abdullah	Ms. Prabhjot Kaur
Head of the Department	Project Guide Supervisor
University Institute of Computing	University Institute of Computing

ACKNOWLEDGEMENT

We deem it a pleasure to acknowledge our sense of gratitude to our project guide Ms. Prabhjot Kaur under whom we have carried out the project work. His incisive and objective guidance and timely advice encouraged us with constant flow of energy to continue the work.

We wish to reciprocate in full measure the kindness shown by Dr. Abdullah (H.O.D, University Institute of Computing) who inspired us with his valuable suggestions in successfully completing the project work.

We shall remain grateful to Dr. Manisha Malhotra, Additional Director, University Institute of Technology, for providing us a strong academic atmosphere by enforcing strict discipline to do the project work with utmost concentration and dedication.

Finally, we must say that no height is ever achieved without some sacrifices made at some end and it is here where we owe our special debt to our parents and our friends for showing their generous love and care throughout the entire period of time.

Date: 29.10.2024

Place: Chandigarh University, Mohali, Punjab

Pankaj Kumar Singh(UID- 24MCA20067)

ABSTRACT

This section provides a high-level overview of the project. The **Simple Calculator** project is a command-line application created in Python, capable of performing fundamental arithmetic operations like addition, subtraction, multiplication, and division. Designed to run on a Linux Fedora system, the calculator emphasizes simplicity and functionality. Key programming concepts, including functions, loops, error handling, and input validation, are incorporated to ensure a smooth user experience and accurate calculations. This project serves as an introductory step for learning Python and Linux command-line operations and is suitable for beginners.

Table of Contents

Introduction

In this section, we introduce the **motivation and purpose** of the project:

Calculators are fundamental tools that help in various everyday activities, from financial calculations to educational tasks. Building a calculator using Python on Linux Fedora provides a hands-on opportunity to apply core programming skills. This project introduces new Python learners to practical coding concepts such as function creation, user interaction, and error management, all while using a terminal-based approach. Additionally, implementing this project on a Linux environment, specifically Fedora, introduces users to command-line utilities, enhancing their comfort with Linux systems.

System Requirements

The **System Requirements** section lists both the hardware and software needs to set up and run the Simple Calculator on a Fedora Linux system.

Hardware Requirements

- **Processor**: An Intel i3 or higher processor provides enough power to run Python scripts smoothly.
- **RAM**: At least 2GB of RAM ensures adequate memory for handling multiple processes during calculations.
- **Disk Space**: The project itself is lightweight, requiring less than 10 MB, but additional space may be needed for related software installations.

Software Requirements

- **Operating System**: Fedora Linux 33 or later versions are compatible, as they come with Python 3 pre-installed or can easily support Python installation.

- **Python**: Version 3.6 or above is recommended for compatibility with syntax and features used in the code.
- **Text Editor**: Editors like Nano, Vim, or Visual Studio Code allow users to write and edit Python code in Fedora.
- **Terminal**: The terminal is used to run the Python script, allowing users to interact with the calculator directly from the command line.

Project Objectives

The **Project Objectives** outline the primary goals:

- **Design a Simple Calculator**: Create a user-friendly command-line calculator for basic arithmetic operations.
- **Introduce Python Fundamentals**: The project demonstrates how to define and call functions, use loops, and handle basic errors.
- **Improve User Interaction**: Implement error messages and prompts to guide the user in performing calculations accurately.
- **Deployment on Linux Fedora**: Deploying the calculator on Fedora allows users to become familiar with Python programming on a Linux platform.

Implementation

Project Setup

The setup instructions involve creating a project directory and initializing the Python script:

- Make sure Python is installed on your Fedora system:

```
python3 --version
```

- If Python is not installed, you can install it with:

```
sudo dnf install python3
```

1. **Directory Creation**: Users create a SimpleCalculator directory to store all project files, making it easier to organize and access files.

```
mkdir SimpleCalculator cd

SimpleCalculator
```

2.Create Python Script: The calculator.py file will contain all code for the calculator application.

```
touch calculator.py
```

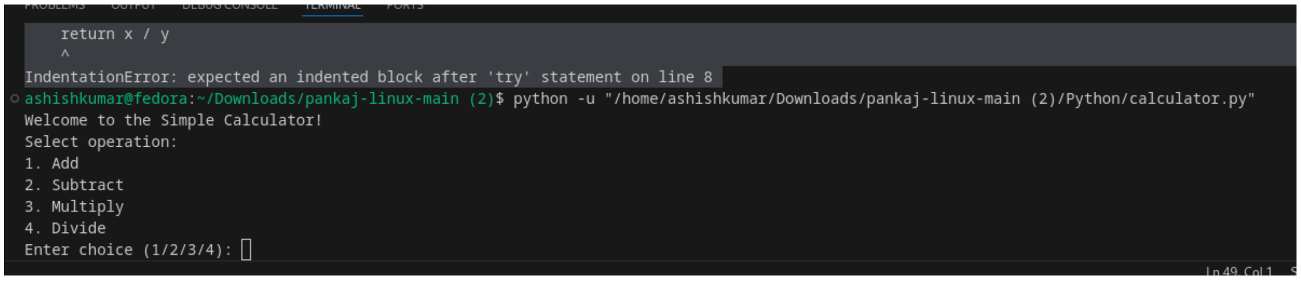
3.Write the Calculator Code: Open `calculator.py` in a text editor and add the following code: # Simple Calculator in Python

```
def add(x, y):
return x + y

def subtract(x, y):
   return x - y

def multiply(x, y):
   return x * y

def divide(x, y):
try:
      return x / y     except
ZeroDivisionError:
      return "Error! Division by zero."

def calculator():
    print("Welcome to the Simple Calculator!")
print("Select operation:")     print("1. Add")
    print("2. Subtract")
print("3. Multiply")     print("4.
Divide")

    while True:
        choice = input("Enter choice (1/2/3/4): ")
         if choice in ('1', '2', '3',
'4'):             try:
                num1 = float(input("Enter first number: "))
num2 = float(input("Enter second number: "))
except ValueError:
                print("Invalid input! Please enter a number.")
continue             if choice == '1':
                print(f"The result is: {add(num1, num2)}")
elif choice == '2':
                print(f"The result is: {subtract(num1, num2)}")
elif choice == '3':
                print(f"The result is: {multiply(num1, num2)}")
elif choice == '4':
                print(f"The result is: {divide(num1, num2)}")

else:
            print("Invalid Input")

        next_calculation = input("Do you want to perform another calculation? (yes/no): ")
if next_calculation.lower() != 'yes':
            break
if __name__ == "__main__":
    calculator()
```

4. Run the Calculator

In the terminal, run your calculator script with:

python3 calculator.py



```
    return x / y
    ^
IndentationError: expected an indented block after 'try' statement on line 8
ashishkumar@fedora:~/Downloads/pankaj-linux-main (2)$ python -u "/home/ashishkumar/Downloads/pankaj-linux-main (2)/Python/calculator.py"
Welcome to the Simple Calculator!
Select operation:
1. Add
2. Subtract
3. Multiply
4. Divide
Enter choice (1/2/3/4): 
```

Code Structure:

The code structure introduces each component of the calculator:

- **Addition (add)**: Takes two numbers and returns their sum.
- **Subtraction (subtract)**: Takes two numbers and returns their difference.
- **Multiplication (multiply)**: Takes two numbers and returns their product.
- **Division (divide)**: Divides two numbers and includes error handling to manage division by zero.
- **Calculator Function (calculator)**: This function handles user input, guides the user in selecting an operation, processes numbers, and displays results. It also provides an option to perform additional calculations or exit the program.

Functional Explanation:

The calculator.py script follows these steps:

1. **Display Menu**: Presents a welcome message and lists available operations.
2. **Operation Selection**: The user selects an operation by entering a number corresponding to a choice (1 for addition, 2 for subtraction, etc.).
3. **Input Handling**: If the user's choice is invalid, they are prompted to enter a valid choice.

4. **Perform Operation**: For a valid choice, the program asks for two numeric inputs and performs the selected operation.
5. **Display Result**: Outputs the result of the calculation to the user.
6. **Repeat or Exit**: Prompts the user to either continue with another calculation or exit the program.

Features:

This section highlights the **features** of the calculator:

- **Arithmetic Operations**: Supports the four primary arithmetic operations.
- **User Guidance**: Shows instructions to guide users through each calculation.
- **Error Management**: Ensures that division by zero is managed gracefully, displaying an error message if an attempt is made.
- **Loop Functionality**: A while loop allows the user to conduct multiple calculations without restarting the program.

Testing and Validation:

Testing ensures the calculator functions as expected:

1. **Valid Inputs**: Basic operations are tested with both positive and negative numbers.
2. **Invalid Inputs**: Edge cases are checked, such as entering text for numbers, choosing invalid options, and dividing by zero.
3. **Result Verification**: Example test cases verify the calculator’s accuracy:
    - **Addition**: 5 + 3 outputs 8. ○ **Subtraction**: 10 - 5 outputs 5.
    - **Division by Zero**: Attempting 10 / 0 results in an error message.

Limitations:

The **Limitations** section describes current constraints:

- **Basic Functionality Only**: The calculator currently supports only addition, subtraction, multiplication, and division. Advanced calculations like exponents or square roots are not included.
- **No GUI**: The application is command-line only, making it less accessible for users who prefer graphical interfaces.
- **Single Number Input**: It does not support calculations with more than two numbers in a single operation.

Future Enhancements:

Possible **Future Enhancements** include:

- **Advanced Functions**: Expanding to include mathematical functions like exponentiation, square roots, and trigonometric calculations.
- **Graphical Interface**: A GUI built with Tkinter or PyQt to make it more user-friendly.
- **Calculation History**: Implement a feature to log previous calculations or export results to a file.

Conclusion:

In **Conclusion**, we summarize the project's success in meeting its objectives. The Simple Calculator on Fedora Linux offers an effective solution for performing basic arithmetic operations. By completing this project, users gain hands-on experience with Python functions, error handling, and terminal-based programming. This project lays a strong foundation for further exploration in Python programming, Linux command-line usage, and more advanced application development.

References

1. **Python Software Foundation. (2023).** *Python 3.10.0 Documentation.* Retrieved from https://docs.python.org/3/ ○ This documentation provides comprehensive guidance on Python functions, loops, and basic syntax, which are essential for creating a calculator application.

2. **Fedora Project. (2023).** *Fedora Linux 33 Documentation.* Retrieved from https://docs.fedoraproject.org/en-US/fedora/ ○ The official Fedora documentation offers resources for installing software, managing files, and using the Fedora terminal, which are helpful when setting up and running Python scripts on Fedora Linux.
3. **Grinberg, M. (2019).** *Fluent Python: Clear, Concise, and Effective Programming.* O'Reilly Media. ○ This book provides an in-depth exploration of Python's advanced features, including handling errors and creating user-friendly command-line applications.
4. **McKinney, W. (2017).** *Python for Data Analysis.* O'Reilly Media.
    - Though focused on data analysis, this book includes useful sections on error handling and managing user input, both relevant for developing robust Python applications.
5. **Severance, C. (2016).** *Python for Everybody: Exploring Data Using Python 3.* Charles Severance. ○ A great beginner's guide to Python programming, covering essential concepts such as functions, loops, and basic data types, which are fundamental for the calculator project.
6. **University of Michigan. (n.d.).** *Python Basics Course (Coursera).* Retrieved from https://www.coursera.org/learn/python-basics ○ This online course introduces Python basics, including building simple applications, managing user inputs, and handling arithmetic operations.
7. **Linux Command Library. (2023).** *Linux Commands – A Practical Guide.* Retrieved from https://www.linuxcommand.org/ ○ Provides an extensive list of Linux commands, including how to use the terminal and run scripts, essential for setting up and executing the Python calculator on Fedora.
8. **Raymond, E. S. (2003).** *The Art of Unix Programming.* Addison-Wesley.
    - This book offers insights into command-line programming principles, which can be applied to enhance the user experience and error handling in a terminal-based calculator.

9. **Shaw, Z. A. (2017).** *Learn Python 3 the Hard Way: A Very Simple Introduction to the Terrifyingly Beautiful World of Computers and Code.* Addison-Wesley.
   - An excellent resource for learning Python by building projects, it includes sections on function creation, loops, and exception handling, which are relevant to building a calculator.
10. **Stack Overflow. (n.d.).** *Python – Error Handling and Exception Management.* Retrieved from https://stackoverflow.com/ ○ Stack Overflow is a useful resource for troubleshooting and finding solutions to common coding errors, such as handling invalid input and division by zero in Python.