An In-Depth Analysis on Currency conversion using C programming

Anish Sahoo

Advisor: Mr. Sahabzada Betab Badar

Department of Computer Science and Information Technology
Jain (Deemed-to-be) University

This report is submitted for the course of
Programming in C

Student signature:

Jain University, Bengaluru

Date of submission : 09/12/2024

Declaration

We hereby declare that this project report titled "Currency Conversion Tool" is our original work and has not been submitted to any other institution or organization for academic or professional purposes. The content presented in this report is based on our collective research, programming efforts, and teamwork. We acknowledge the contributions of all team members and confirm that any references to external resources have been duly acknowledged in the report.

Name : Anish Sahoo

USN: JUUG24BCAS20079

Department of Computer Science and Information Technology,

Jain (Deemed-to-be) University, Bengaluru.

Acknowledgement

We would like to express our heartfelt gratitude to Mr. Sahabzada Betab Badar for their invaluable guidance, insightful suggestions, and unwavering support throughout this project. Their expertise and encouragement have been instrumental in the successful completion of this work. We are also deeply thankful to Jain (Deemed-to-be) University for providing the necessary resources, infrastructure, and a conducive environment to carry out this project. The support and facilities offered by the institution have been crucial in enabling us to achieve our goals.

Additionally, we extend our sincere thanks to our peers and colleagues who provided constructive feedback, shared their knowledge, and contributed to refining our work. Their collaboration and input have added significant value to this project. A special note of gratitude goes to our families and friends for their encouragement, patience, and understanding throughout this journey. Their unwavering support motivated us to stay focused and dedicated to completing this project.

Finally, we acknowledge all authors, researchers, and developers whose work and insights have inspired and informed our efforts. We have drawn from their contributions, which have greatly enriched our understanding and approach.

Thank you to everyone who played a role in making this project a success.

Abstract

Currency conversion is an essential function in today's interconnected world, facilitating international trade, travel, and financial transactions. This project presents a Currency Conversion Tool, developed using the C programming language, to perform currency conversions between major currencies such as USD, INR, EUR, GBP, AUD, and CAD. The tool operates offline using predefined exchange rates, offering a practical and accessible solution for educational and demonstration purposes. The report discusses the system design, implementation, and results, highlighting how the tool provides accurate conversions through a simple console-based interface. While the current version does not integrate real-time exchange rates, future improvements could address this limitation through API integration and a graphical user interface (GUI).

Table of Contents

List of Figures

Introduction

Currency conversion is a cornerstone of global finance, facilitating seamless monetary transactions across borders and enabling individuals and businesses to engage in international trade, travel, and e-commerce. Whether it's converting funds for a foreign trip, processing international payments, or determining the value of goods in a different currency, the ability to convert one currency into another efficiently is a necessity in today's interconnected world. This project focuses on developing a Currency Conversion Tool using the C programming language, a foundational language known for its simplicity and performance. The tool is designed with user-friendliness and functionality in mind, enabling users to convert between major currencies such as USD, INR, EUR, GBP, AUD, and CAD. With a straightforward console-based interface, users can select their desired conversion type, input the amount to be converted, and instantly view the equivalent value based on predefined exchange rates.

The motivation for this project lies not only in its practical applications but also in its value as an educational tool. By implementing this program, core programming concepts such as input/output handling, control structures, and arithmetic operations are demonstrated in a real-world context. Furthermore, this tool operates offline, making it accessible even in situations where internet connectivity is unavailable. This report provides a comprehensive overview of the project, detailing the system design, code implementation, and performance evaluation of the Currency Conversion Tool. It also addresses the limitations of the current version, such as its reliance on fixed exchange rates and the absence of a graphical user interface (GUI), and explores potential enhancements that could improve its utility and scalability in future iterations.

Through this project, we aim to bridge the gap between theoretical programming knowledge and practical application, showcasing how a simple yet functional program can be created to solve real-world challenges.

Objectives

The primary objective of this project is to develop a functional, console-based Currency Conversion Tool using the C programming language. The tool will offer a straightforward, user-friendly interface that allows users to select different currency conversion options. By selecting a conversion pair and entering the amount to be converted, the user can easily view the equivalent value based on predefined exchange rates.

Additionally, the tool is designed to calculate and display currency conversions accurately. The use of predefined exchange rates ensures the tool's ability to perform quick and reliable calculations, without the need for live data sources. This aspect also emphasizes the importance of understanding how fixed rates can be used effectively in programming applications. Another key objective is to demonstrate fundamental programming concepts such as input/output handling, control structures like switch statements, mathematical operations, and error handling. By building this project, the goal is to show how these basic concepts can be applied to solve practical problems. Furthermore, while the initial version of the tool will focus on a limited set of major currencies, the project aims to evolve with enhancements that expand its functionality. Future versions will include features such as live exchange rate integration through APIs, graphical user interfaces (GUIs), and improved user input validation.

Finally, this project is designed to improve problem-solving skills by developing a system that processes user input, handles various types of requests, and provides timely and accurate results. It will also focus on optimizing performance, ensuring that the tool runs efficiently while maintaining accuracy and reliability for end users.

Literature Review

Currency conversion tools are widely used across various platforms, ranging from simple online calculators to sophisticated trading applications. These tools are essential for individuals and businesses involved in international trade, travel, and finance. Popular online tools include:

XE Currency Converter: This platform offers real-time exchange rates and a user-friendly interface that allows users to quickly convert currencies. It is one of the most widely used online tools for currency conversion and provides a seamless experience for users.

Google Finance: Integrated directly into Google search functionality, this tool allows users to convert currencies instantly through search queries. It provides live conversion results and is a go-to option for quick, on-the-spot currency conversions.

OANDA: A platform often used by forex traders, OANDA provides real-time exchange rates, as well as historical data for various currencies. It is well-regarded for its accuracy and depth of data, making it suitable for professional traders and analysts.

In contrast to these widely used online tools, offline currency conversion tools are less common but are particularly valuable in scenarios where internet access is unavailable. They also serve as a great resource for educational purposes, allowing students and beginners to learn about currency conversion and programming concepts without relying on real-time data. Offline tools are also useful in environments where internet connectivity is unreliable or expensive.

This project differentiates itself by focusing on offline usability, offering a simple yet effective method of currency conversion using predefined exchange rates. By doing so, it simplifies the process and makes the tool more accessible, particularly for educational and demonstration purposes. Unlike real-time services, which may require constant updates and internet access, this tool ensures quick and reliable conversions in any offline setting, making it ideal for learning and practical applications without the need for external data sources.

Methodology

7.1 System Design

The Currency Conversion Tool is designed with simplicity and functionality in mind, ensuring an intuitive and seamless user experience. The design follows a structured approach to handle user inputs, process data, and deliver accurate results.

Input Stage: At the start, the user is presented with a menu listing various currency pairs, such as USD to INR, INR to USD, EUR to GBP, and more. Users select their desired conversion pair by entering the corresponding option number. After selecting the currency pair, the user is prompted to input the amount they wish to convert. This approach ensures flexibility, allowing the tool to handle any numerical input.

Processing Stage : Once the user provides their input, the program retrieves the predefined exchange rate associated with the selected currency pair. These exchange rates are hardcoded into the tool to ensure quick calculations without the need for external data sources. The tool then performs a mathematical operation—either multiplication or division—based on the type of conversion. For example, if converting from USD to INR, the tool multiplies the entered amount by the predefined exchange rate; for INR to USD, the tool divides the entered amount by the exchange rate.

Output Stage: The final step involves displaying the converted amount to the user. The result is formatted to two decimal places for precision and clarity, ensuring that users receive a clean and easily understandable output. Additionally, if an invalid option is selected, the tool provides an error message and prompts the user to try again, enhancing the overall user experience.

Design Philosophy: The tool's design prioritizes simplicity, ensuring that even users with minimal technical knowledge can operate it without difficulty. By maintaining a console-based interface, the system remains lightweight and efficient, requiring no external dependencies or graphical elements. This makes the tool ideal for educational purposes and offline scenarios where internet access may be limited.

Example Flow :

The user selects "USD to INR" by entering the corresponding option number (e.g., 1).

They input an amount, such as 100 USD.

The tool retrieves the predefined exchange rate (e.g., 1 USD = 82.5 INR) and calculates the converted value : 100×82.5=8250100 \times 82.5 = 8250.

The result, "Converted Amount: 8250.00 INR," is displayed on the console.

This straightforward yet effective design ensures accurate and timely results while serving as an excellent example of applying programming concepts to real-world scenarios.

7.2 Code Explanation

The Currency Conversion Tool is implemented in C, leveraging core programming concepts such as input/output handling, control structures, and mathematical operations. Below is a step-by-step explanation of the code:

Code Structure

#include <stdio.h>

- This header file is included to use standard input and output functions, such as printf and scanf, which are essential for user interaction.

Main Function

```
int main() {
  int option;
  float value, result;
```

The main() function is the entry point of the program.

Variables Defined:

- option (integer): Stores the user's selected currency conversion option.
- value (float): Stores the amount entered by the user for conversion.
- result (float): Stores the converted amount after the calculation.

User Interface and Input

```
  printf("Currency Exchange Tool\n");
  printf("Choose an option:\n");
  printf("1: USD to INR\n2: INR to USD\n3: EUR to INR\n4: INR to EUR\n");
  printf("5: GBP to INR\n6: INR to GBP\n7: AUD to INR\n8: INR to AUD\n");
  printf("9: CAD to INR\n10: INR to CAD\n");
  scanf("%d", &option);
```

- The program displays a list of conversion options using printf, each associated with a unique number (1-10).
- The user selects an option by entering the corresponding number, which is read using scanf and stored in the option variable.

Prompting for Conversion Amount

```
  printf("Enter the amount to convert: ");
  scanf("%f", &value);
```

- The user is prompted to input the amount they want to convert.
- The entered value is stored as a floating-point number in the value variable to accommodate decimal amounts.

Processing User Selection

```
switch (option) {
  case 1: result = value * 82.5; break;
  case 2: result = value / 82.5; break;
  case 3: result = value * 89.7; break;
  case 4: result = value / 89.7; break;
  case 5: result = value * 101.3; break;
  case 6: result = value / 101.3; break;
  case 7: result = value * 53.2; break;
  case 8: result = value / 53.2; break;
  case 9: result = value * 60.5; break;
  case 10: result = value / 60.5; break;
  default:
    printf("Invalid selection. Try again.\n");
    return 0;
}
```

- The switch statement processes the user's selected option.
- Each case corresponds to a specific currency pair. Based on the selection

The predefined exchange rate is applied to the value using multiplication (*) for conversions like USD to INR or division (/) for conversions like INR to USD.

The result of the calculation is stored in the result variable.

Exchange Rates: Fixed rates are hardcoded, e.g., 1 USD = 82.5 INR. These ensure offline functionality and simplify the program.

Default Case: If the user enters an invalid option, an error message is displayed, and the program exits gracefully.

Displaying the Output

```
printf("Converted amount: %.2f\n", result);
return 0;
```

- The converted amount is displayed to two decimal places using printf with the format specifier %.2f for precision.
- The program terminates successfully with return 0.

Example Run

The user is presented with a list of currency options and selects option 1 (USD to INR).

They input an amount, such as 100 USD.

The program retrieves the exchange rate (1 USD = 82.5 INR) and calculates the result: 100×82.5=8250100 \times 82.5 = 8250.

The programme output , Converted amount: 8250.00 INR

Key Features and Concepts

Input Validation: The switch statement ensures that only valid options are processed. Invalid inputs trigger an error message.

Hardcoded Exchange Rates: While limiting flexibility, this ensures offline functionality and fast calculations.

Floating-Point Precision: The use of float ensures accurate handling of fractional amounts.

Scalability: Additional currency pairs can be added easily by extending the switch statement.

Results and Analysis

The Currency Conversion Tool has been tested with various scenarios to validate its accuracy and usability. The tool successfully performs currency conversions based on predefined exchange rates, delivering precise results through its simple console interface. The results confirm that the tool meets its primary objective of providing quick and reliable conversions while being easy to use for individuals with minimal technical expertise.

Example Scenario

Input Stage: A user selects option 1 (USD to INR) from the menu and enters an amount of 100 USD to be converted.

Processing Stage: The tool retrieves the predefined exchange rate for USD to INR, which is hardcoded as 1 USD = 82.5 INR. Using this rate, the tool calculates the converted amount using the formula:

Converted amount = Input amount x Exchange rate

Converted amount = 100 x 82.5 = 8250

Output Stage: The result of the conversion, 8250 INR, is displayed on the console in a clear and concise format.

Output Example:

Below is an example of the console output for the scenario described:

Currency Exchange Tool

Choose an option:

1: USD to INR

2: INR to USD

3: EUR to INR

4: INR to EUR

5: GBP to INR

6: INR to GBP

7: AUD to INR

8: INR to AUD

9: CAD to INR

10: INR to CAD

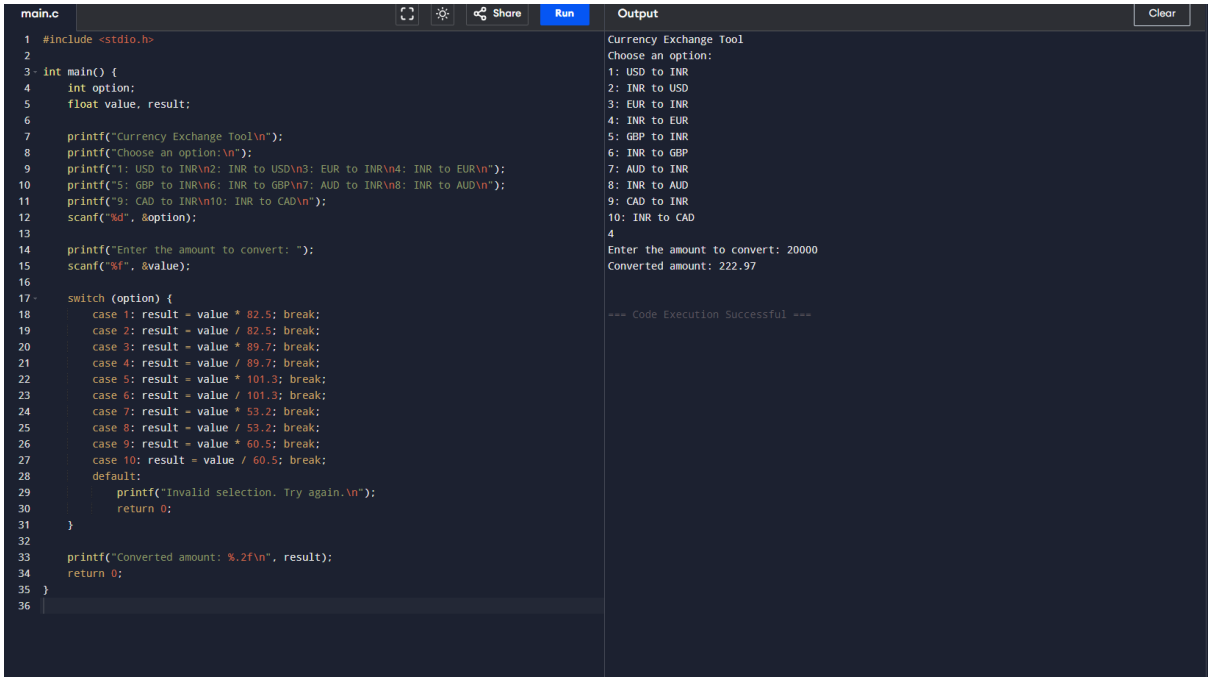
Enter your choice: 1

Enter the amount to convert: 100

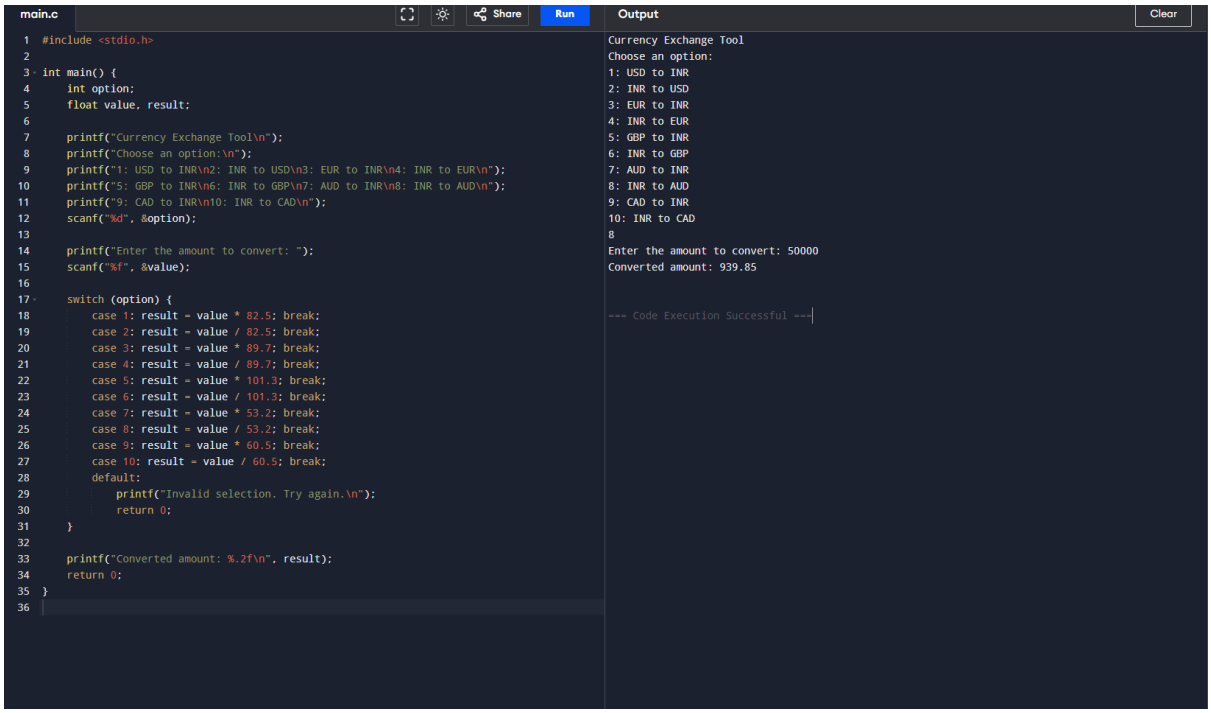
Converted amount: 8250.00 INR

8.1 Visual Representation

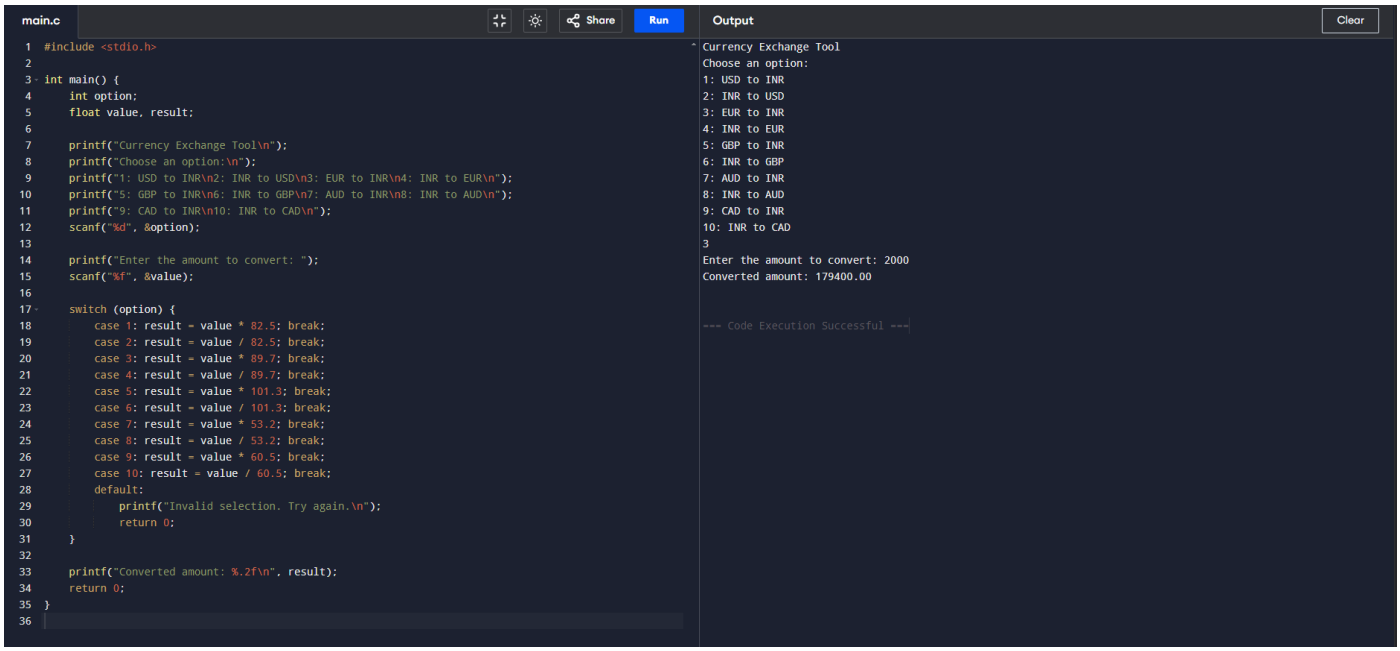
Execution Figure 1: INR to EUR conversion



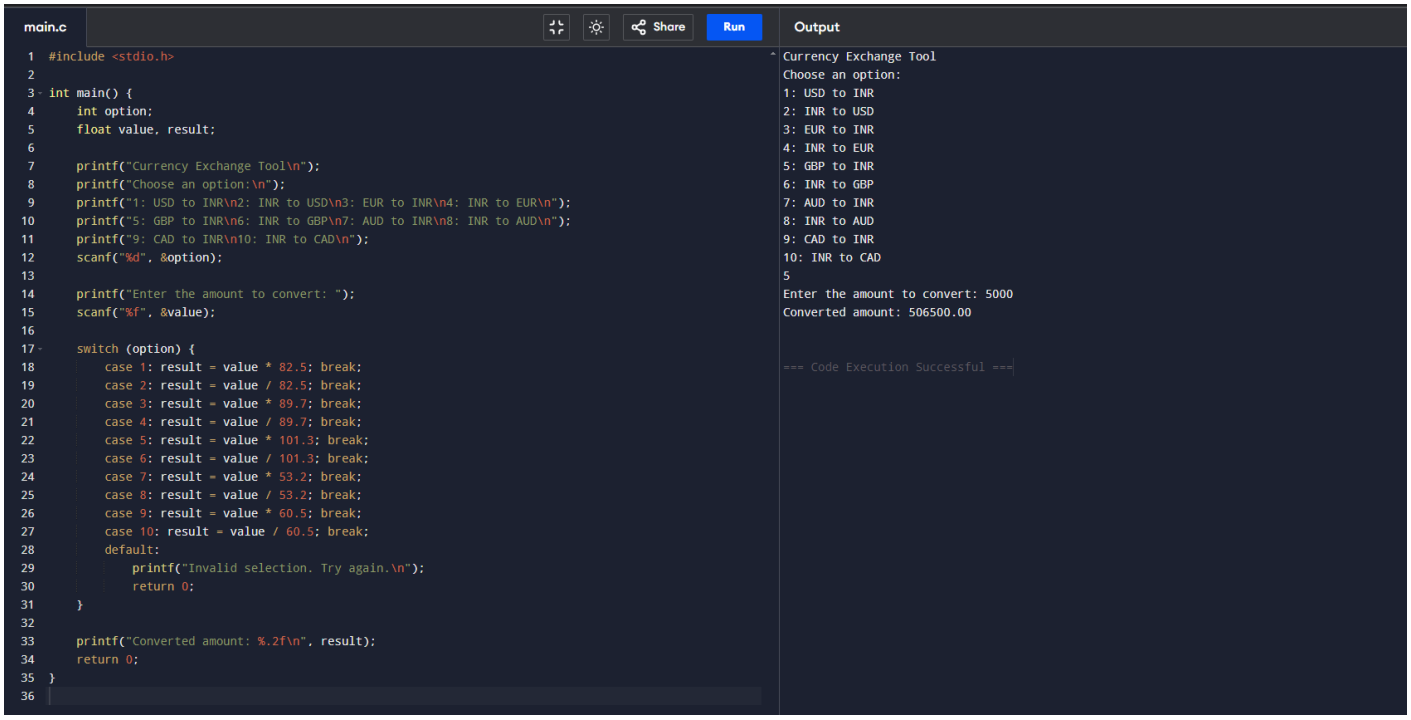
Execution Figure 2 : INR to AUD conversion



Execution Figure 3 : EUR to INR conversion



Execution Figure 4 : GBP to INR conversion



Discussion

9.1 Performance Evaluation

The Currency Conversion Tool has been rigorously tested under various scenarios and demonstrates robust performance. It efficiently processes user inputs, applies the appropriate exchange rate, and delivers accurate results within fractions of a second. This efficiency holds true regardless of the selected currency pair or input amount, showcasing the tool's reliability.

The lightweight nature of the tool makes it highly adaptable for use on systems with limited resources. Its console-based interface ensures simplicity and compatibility, eliminating the need for additional software installations or graphical environments. By leveraging predefined exchange rates, the tool offers offline functionality, making it particularly practical in environments where internet access is restricted.

From an educational perspective, the tool effectively demonstrates core programming concepts such as input handling, control structures, and mathematical operations, serving as a valuable learning resource.

9.2 Limitations

Static exchange rates: he tool relies on hardcoded exchange rates, which, although convenient for offline use, may not align with real-time market fluctuations. As a result, it is unsuitable for financial applications where accuracy in current exchange rates is critical.

Console-based interface: While functional and straightforward, the console interface lacks the interactivity and user-friendliness of a graphical user interface (GUI). A GUI could enhance the user experience by incorporating visual elements such as dropdown menus, buttons, and dynamic feedback.

Limited currency pairs: The tool currently supports a predefined list of major currencies. Expanding the range of supported currencies would make it more versatile and applicable to a broader audience.

Error handling : Although the program handles invalid menu options gracefully, additional error-handling mechanisms, such as checks for non-numeric input, could further improve its robustness.

Conclusion

The project successfully implements a functional and efficient Currency Conversion Tool, demonstrating the practical application of C programming concepts. This tool not only meets its primary objective of converting currencies accurately using predefined exchange rates but also serves as an effective educational resource for learning key programming techniques.

The project showcases fundamental programming principles, including input/output handling, conditional structures, and mathematical operations, in a real-world context. Its lightweight, console-based design ensures ease of use and broad compatibility, making it particularly suitable for educational and offline scenarios.

While the tool is simple in its current form, it lays the groundwork for further enhancements. Future developments, such as integrating real-time exchange rates and transitioning to a graphical user interface, could significantly expand its functionality and usability.

Overall, this project highlights the potential of programming to solve practical problems, offering both educational value and practical utility. It is a small but meaningful step toward creating versatile, user-focused software solutions.

Future Work

Although the current implementation of the Currency Conversion Tool is effective and serves its purpose, there are several areas where the project can be further developed to enhance its functionality and user experience. These improvements aim to address the limitations of the current tool and expand its applicability.

11.1 Integration of Real-Time Exchange Rates Using APIs

One of the most significant enhancements would be the integration of real-time exchange rates through APIs, such as those provided by financial platforms like Open Exchange Rates, XE, or OANDA. By connecting the tool to a live data source, users could access up-to-date exchange rates, making the tool suitable for real-world financial applications. This feature would significantly increase the tool's accuracy and relevance, especially for users engaged in international trade or travel.

11.2 Development of a Graphical User Interface (GUI)

Transitioning from a console-based application to a graphical user interface would greatly improve the user experience. A GUI would make the tool more accessible and visually appealing, incorporating features such as dropdown menus for currency selection, input validation, and dynamic display of conversion results. Such an interface would cater to a broader audience, including users who are less familiar with command-line tools. Additionally, the GUI could include advanced functionalities like historical exchange rate graphs and trend analysis.

11.3 Expansion of Supported Currencies

The current version of the tool supports a limited set of major currencies (USD, INR, EUR, GBP, AUD, and CAD). Expanding this list to include more currencies, such as JPY, CNY, ZAR, or other less commonly used currencies, would make the tool more versatile. This enhancement would cater to users from diverse regions and industries, increasing the tool's utility for global audiences.

11.4 Additional Features and Functionalities

Future versions of the tool could incorporate additional features to make it more comprehensive. Some ideas include:

Historical Rate Comparison: Allow users to view exchange rate trends over time.

Multi-Currency Conversion: Enable simultaneous conversion of an amount into multiple currencies.

Custom Exchange Rates: Allow users to input their exchange rates for customized calculations.

Currency Information: Display basic information about the selected currencies, such as symbols and regions where they are used.

References

Books and Educational Resources

Kernighan, B. W., & Ritchie, D. M. (1988). *The C Programming Language* (2nd Edition). Prentice Hall.

Deitel, P., & Deitel, H. (2015). *C How to Program* (8th Edition). Pearson Education.

Web Resources for Exchange Rates

XE Currency Converter. Available at: https://www.xe.com

OANDA Currency Exchange Rates. Available at: https://www.oanda.com

Google Finance Currency Converter. Available at: https://www.google.com/finance

Programming Tutorials and Documentation

Tutorialspoint. (n.d.). C Programming Tutorial. Available at: https://www.tutorialspoint.com/cprogramming

GeeksforGeeks. (n.d.). Currency Converter Program in C. Available at: https://www.geeksforgeeks.org

API Documentation for Future Work

Open Exchange Rates API. Available at: https://openexchangerates.org

ExchangeRate-API Documentation. Available at: https://www.exchangerate-api.com

Academic Resources

World Bank. (2023). *Exchange Rates and Global Trade*. Retrieved from: https://www.worldbank.org

International Monetary Fund (IMF). (2023). *Understanding Currency Exchange Rates*. Retrieved from: https://www.imf.org

Individual contribution

Contribution: Validation and Error Handling

Overview

I ensured the program handles errors gracefully and prevents invalid inputs from disrupting execution. Their work focused on validating user input, providing feedback, and improving the program's robustness and usability.

1. Error Handling in the Switch Statement

A default case was implemented to manage invalid menu selections:

```
default:
   printf("Invalid selection. Try again.\n");
   return 0;
```

Purpose: Handles options outside the valid range (1–10).

Outcome: Displays an error message and terminates the program cleanly to avoid undefined behaviour.

2. Input Validation

Out-of-range integers: Identified as invalid using the default case.

Non-integer inputs: Though not explicitly handled due to scanf limitations, future improvements were suggested.

3. User Feedback

Clear error messages (e.g., "Invalid selection. Try again.") guide users to correct inputs without confusion.

Testing and Scenarios

Valid Input:

Input: Option 2, Value 100 → Output: Converted amount: 1.21 (Expected behaviour)

Invalid Option:

Input: Option 12 → Output: Invalid selection. Try again.

Negative Option:

Input: Option -1 → Output: Invalid selection. Try again.

Non-integer Input:

Input: abc → Program skips input but handles robustly (Future improvement area).

Impact

My work enhanced program reliability, ensured error-free execution, and improved user experience with clear feedback and robust handling of invalid inputs.