DESIGN DOCUMENT

1. Title

Project Name: **NUMBERS TO WORDS AND VICE VERSA.**

Authors: **M GOURI SANKAR & GAUTHAM S THAMPI**

Date: ***May 13, 2025***

2. Overview

This project is developed to convert:

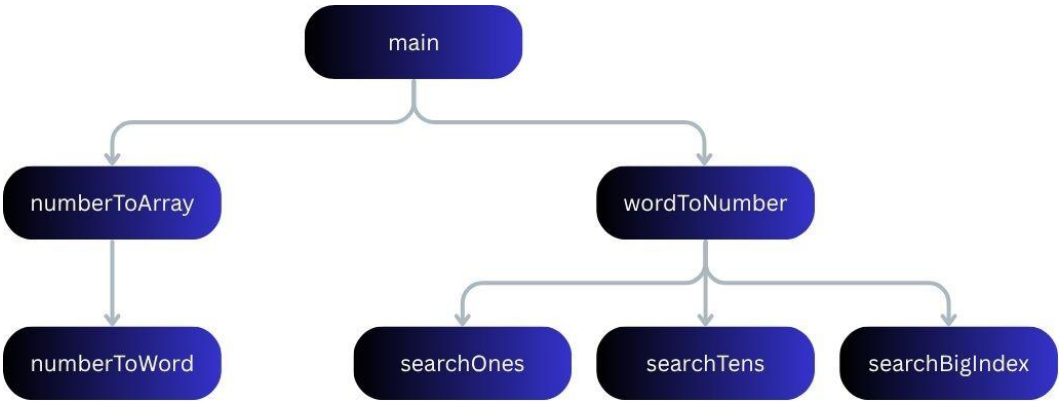
- Numeric values to their word representation (123 → "one hundred and twenty-three"), and
- Words to numeric values ("one hundred" → 100).

3. Problem Statement

- The goal of this project is to build a function that can convert numbers into English words and another function which converts English words back into numbers.
- The function must support **conversion between 1 and 99 crores**, including reverse conversion. For example:

1. 5,67,89,012 → "five crore sixty-seven lakh eighty-nine thousand twelve"
2. "ninety-nine crore" → 99,00,00,000

4. Proposed Solution



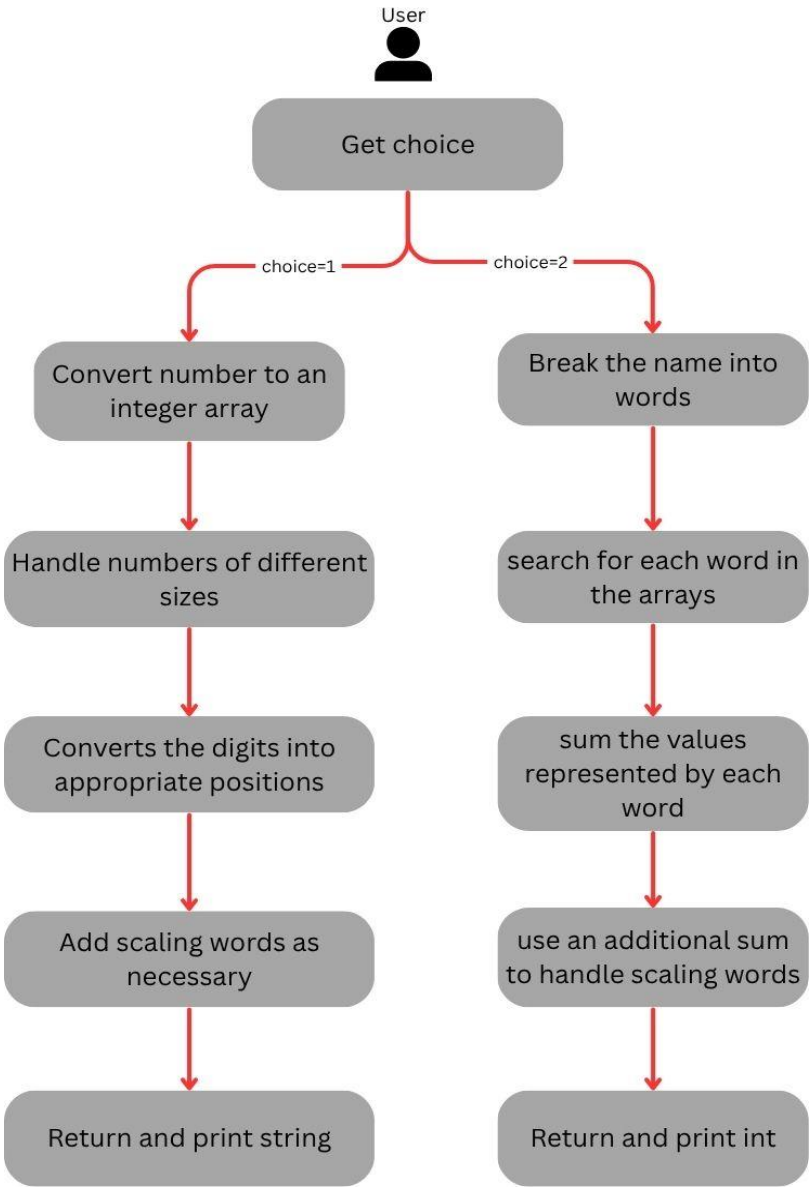
Primary Functions

- char* numberToWord(int[], int, char[]);
- int numberToArray(int[], int);

Helper Functions

- int searchTens(char[]);
- int searchOnes(char[]);
- int searchBigIndex(char[]);
- int wordToNumber(char[]);

5. Logic Breakdown



1. Conversion Direction Handling:
   The program starts by asking the user to choose between two options:
   - Number to word, or
   - Word to number. It routes the input accordingly using a switch-case.
2. Number to Word Conversion:
   - The number is first split into its digits and stored in an array using numberToArray().
   - numberToWord() then maps each digit (left to right) to corresponding words using predefined arrays (ones, tens, elevens) and appends appropriate scale words (thousand, lakh, crore) based on digit position.
   - Special conditions handle hundreds and numbers in the teens.
3. Word to Number Conversion:
   - The input string is tokenized, and each token is searched in predefined arrays to get numeric values.
   - The function wordToNumber() uses searchOnes(), searchTens(), searchElevens(), and searchBigIndex() to determine the numeric equivalent of each word.
   - Scaled values like "lakh" or "crore" multiply accumulated numbers using pow(10, x) to position them correctly.
4. Use of predefined arrays:
   - The arrays ones, tens, elevens, and bigIndex store the word representations of digits, multiples of ten, teen numbers, and scale units.
   - Matching is done by removing spaces before comparison.

5. Numbering Handling:
    - The code follows the Indian numbering system such as lakh,crore etc.
    - Scale multipliers are computed dynamically using position-based logic tied to the bigIndex array.

6. Conclusion

- This system properly converts numbers to words and vice versa, providing a valuable feature for applications such as financial software, voice assistants, and educational tools. The design is modular, scalable, and easy to maintain.