C++ NUMBER SYSTEM CALCULATOR

DOCUMENTATION

BY

ANTHONY GATITU

Converts a decimal number to the specified base.

Parameters

num: The decimal number to be converted. base:

The target base for conversion.

Returns

A string representing the number in the target base.

Details

The function uses a loop to convert the integer part of the decimal number to the target base.

It handles both integer and fractional parts separately, including a specified number of decimal places.

2. Base To Decimal

Converts a number from any base to decimal.

Parameters

num: The number in the source base as a string. base:

The source base of the input number.

Returns

The decimal equivalent of the input number.

Details

The function iterates through the input string, separating the integer and fractional parts.

It uses mathematical operations to calculate the decimal equivalent.

Usage

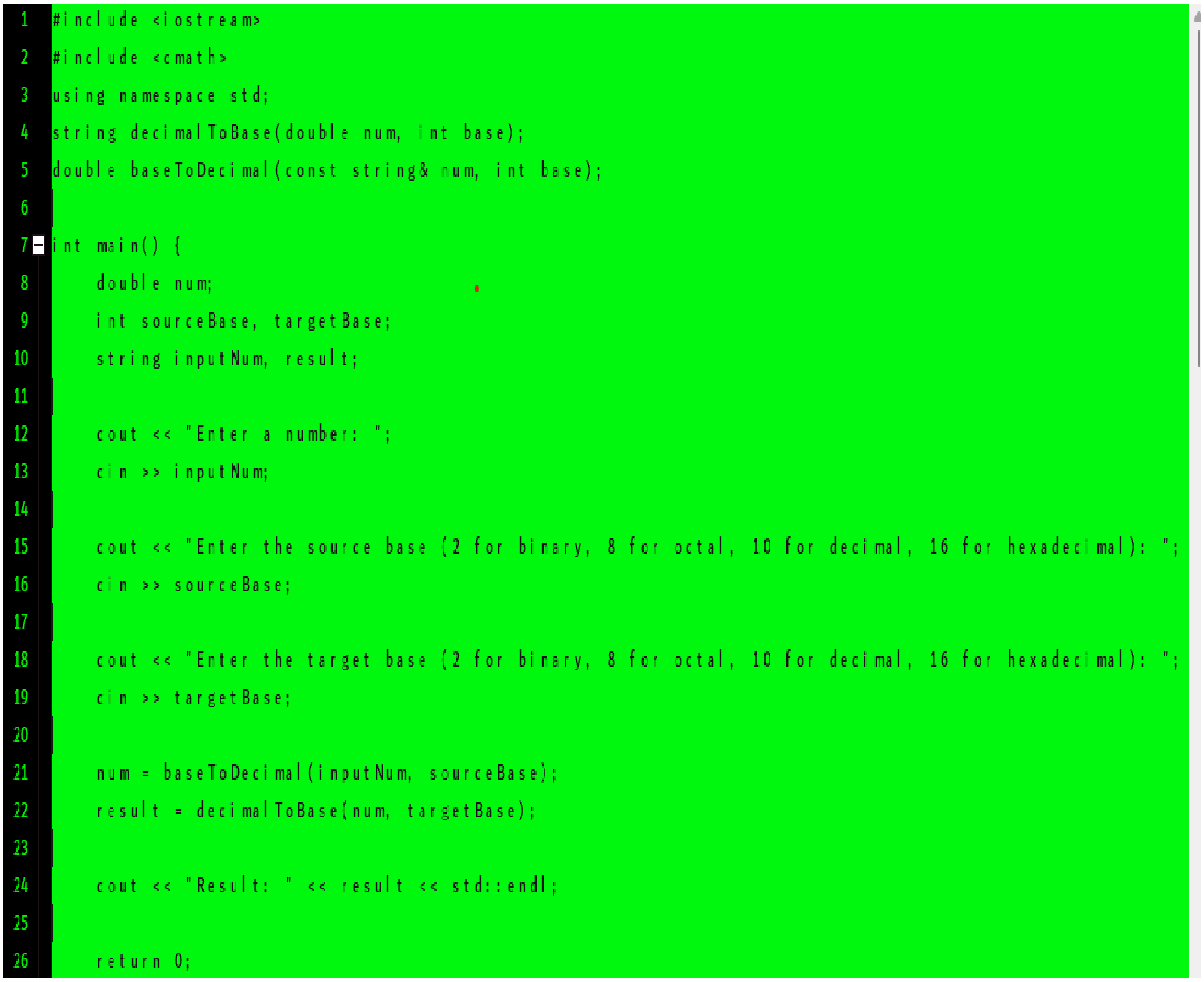
Enter the number to be converted.

Enter the source base (2 for binary, 8 for octal, 10 for decimal, 16 for hexadecimal).

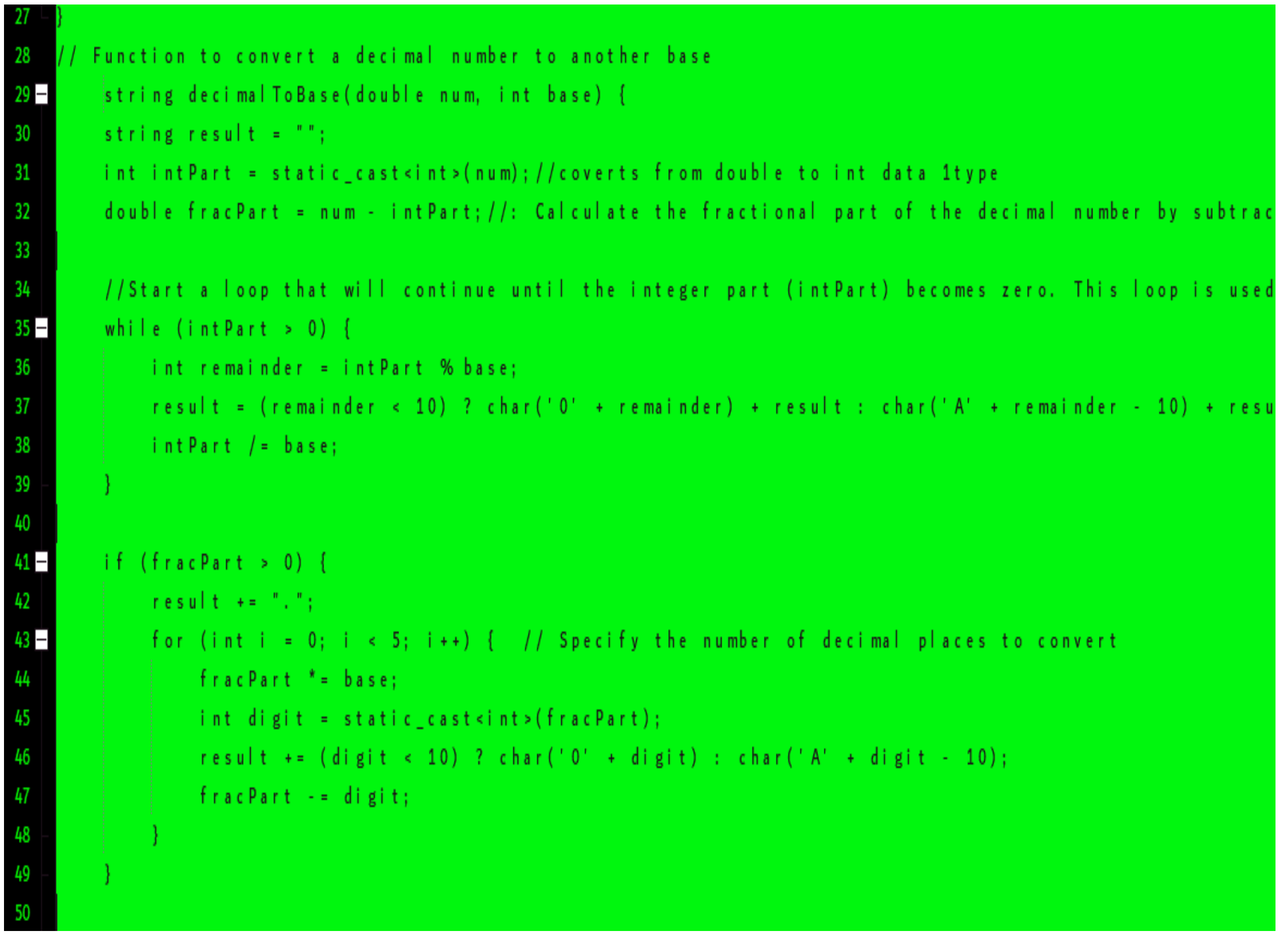
Enter the target base (2 for binary, 8 for octal, 10 for decimal, 16 for hexadecimal).

The program will then display the result of the conversion.

C++ CONVERTER PROGRAM

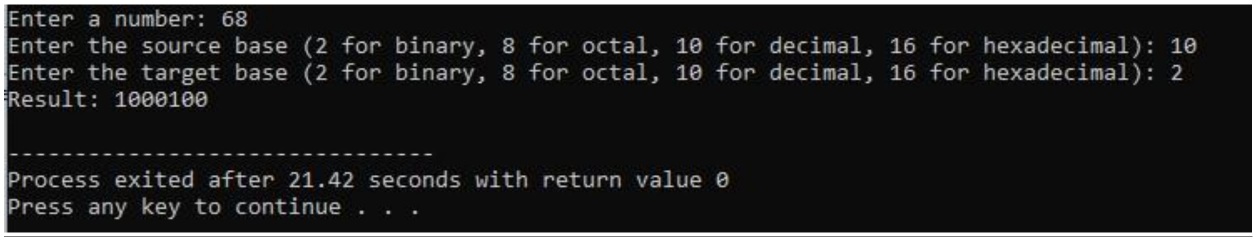


```cpp
#include <iostream>
#include <cmath>
using namespace std;
string decimalToBase(double num, int base);
double baseToDecimal(const string& num, int base);

int main() {
    double num;
    int sourceBase, targetBase;
    string inputNum, result;

    cout << "Enter a number: ";
    cin >> inputNum;

    cout << "Enter the source base (2 for binary, 8 for octal, 10 for decimal, 16 for hexadecimal): ";
    cin >> sourceBase;

    cout << "Enter the target base (2 for binary, 8 for octal, 10 for decimal, 16 for hexadecimal): ";
    cin >> targetBase;

    num = baseToDecimal(inputNum, sourceBase);
    result = decimalToBase(num, targetBase);

    cout << "Result: " << result << std::endl;

    return 0;
```



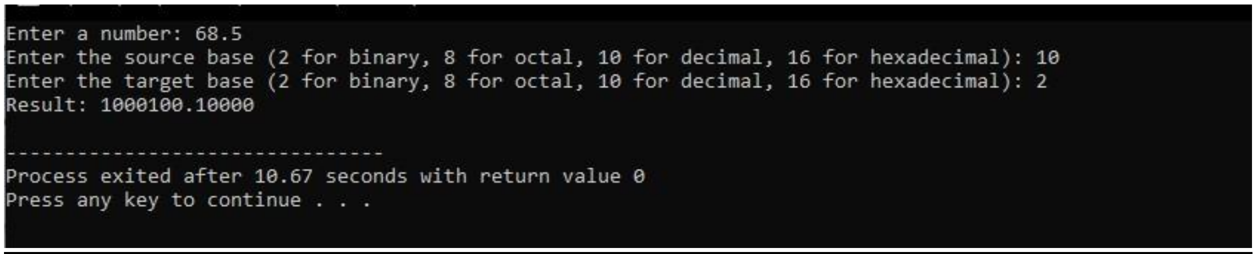
```
}
// Function to convert a decimal number to another base
    string decimalToBase(double num, int base) {
    string result = "";
    int intPart = static_cast<int>(num);//coverts from double to int data 1type
    double fracPart = num - intPart;//: Calculate the fractional part of the decimal number by subtrac

    //Start a loop that will continue until the integer part (intPart) becomes zero. This loop is used
    while (intPart > 0) {
        int remainder = intPart % base;
        result = (remainder < 10) ? char('0' + remainder) + result : char('A' + remainder - 10) + resu
        intPart /= base;
    }

    if (fracPart > 0) {
        result += ".";
        for (int i = 0; i < 5; i++) {  // Specify the number of decimal places to convert
            fracPart *= base;
            int digit = static_cast<int>(fracPart);
            result += (digit < 10) ? char('0' + digit) : char('A' + digit - 10);
            fracPart -= digit;
        }
    }

```

CODE SNIPPET CONVERTING FROM DECIMAL TO BINARY



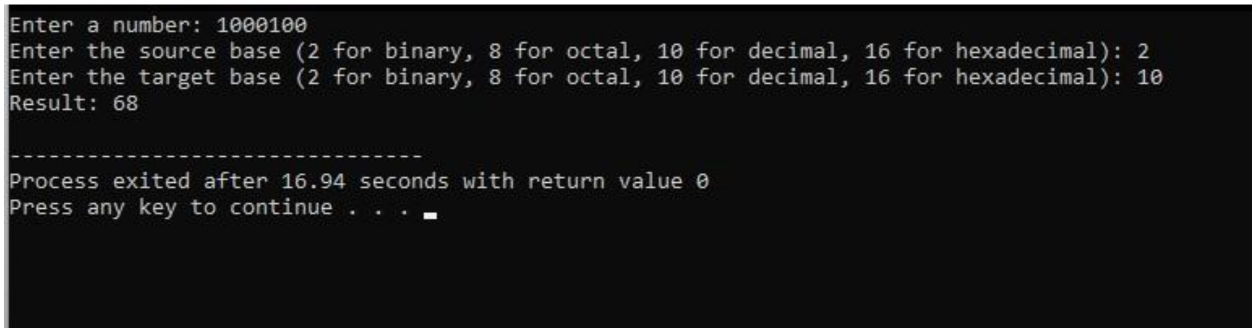
```
Enter a number: 68
Enter the source base (2 for binary, 8 for octal, 10 for decimal, 16 for hexadecimal): 10
Enter the target base (2 for binary, 8 for octal, 10 for decimal, 16 for hexadecimal): 2
Result: 1000100

--------------------------------
Process exited after 21.42 seconds with return value 0
Press any key to continue . . .
```

FRACTIONAL PART CONVERTION

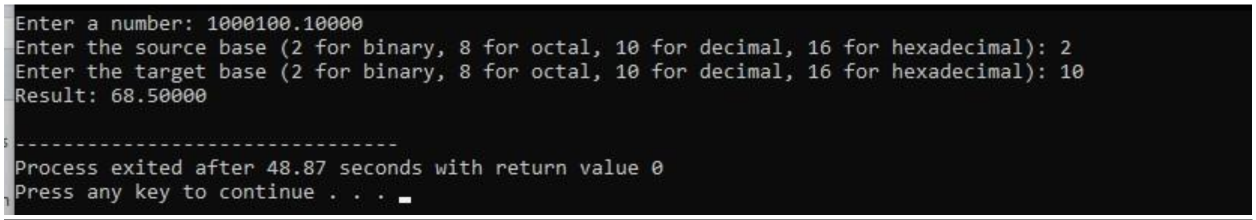


```
Enter a number: 68.5
Enter the source base (2 for binary, 8 for octal, 10 for decimal, 16 for hexadecimal): 10
Enter the target base (2 for binary, 8 for octal, 10 for decimal, 16 for hexadecimal): 2
Result: 1000100.10000

--------------------------------
Process exited after 10.67 seconds with return value 0
Press any key to continue . . .
```

CODE SNIPPET CONVERTING FROM DECIMAL TO BINARY

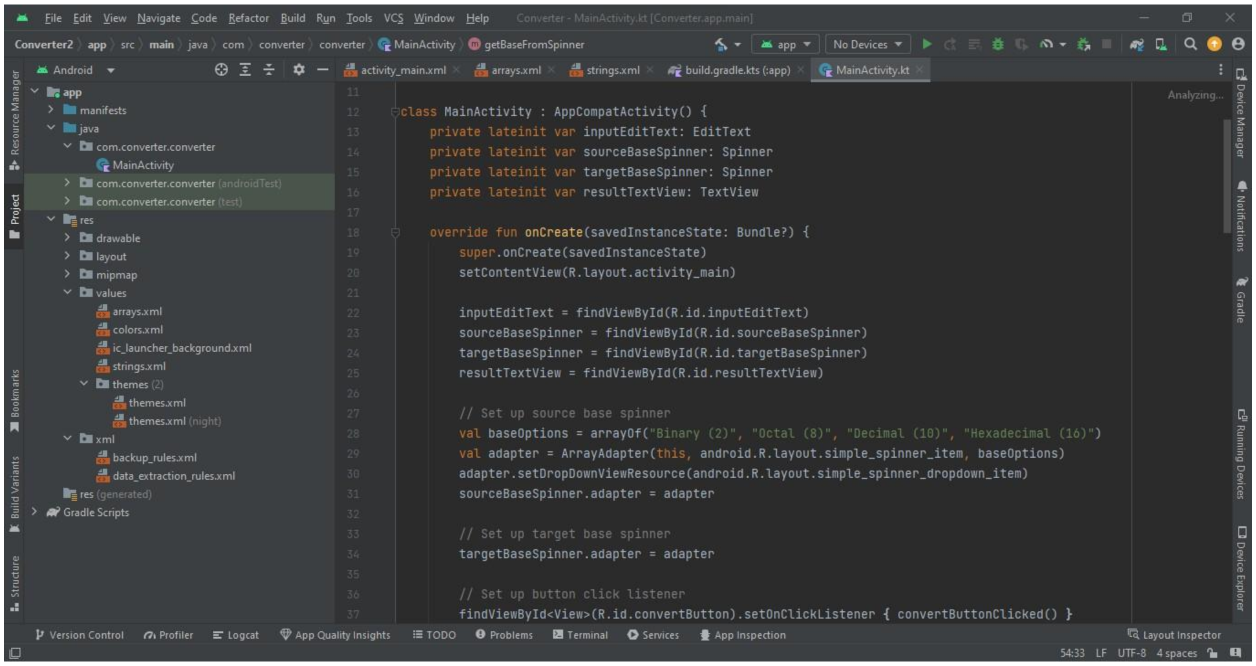


```
Enter a number: 1000100
Enter the source base (2 for binary, 8 for octal, 10 for decimal, 16 for hexadecimal): 2
Enter the target base (2 for binary, 8 for octal, 10 for decimal, 16 for hexadecimal): 10
Result: 68

--------------------------------
Process exited after 16.94 seconds with return value 0
Press any key to continue . . .
```

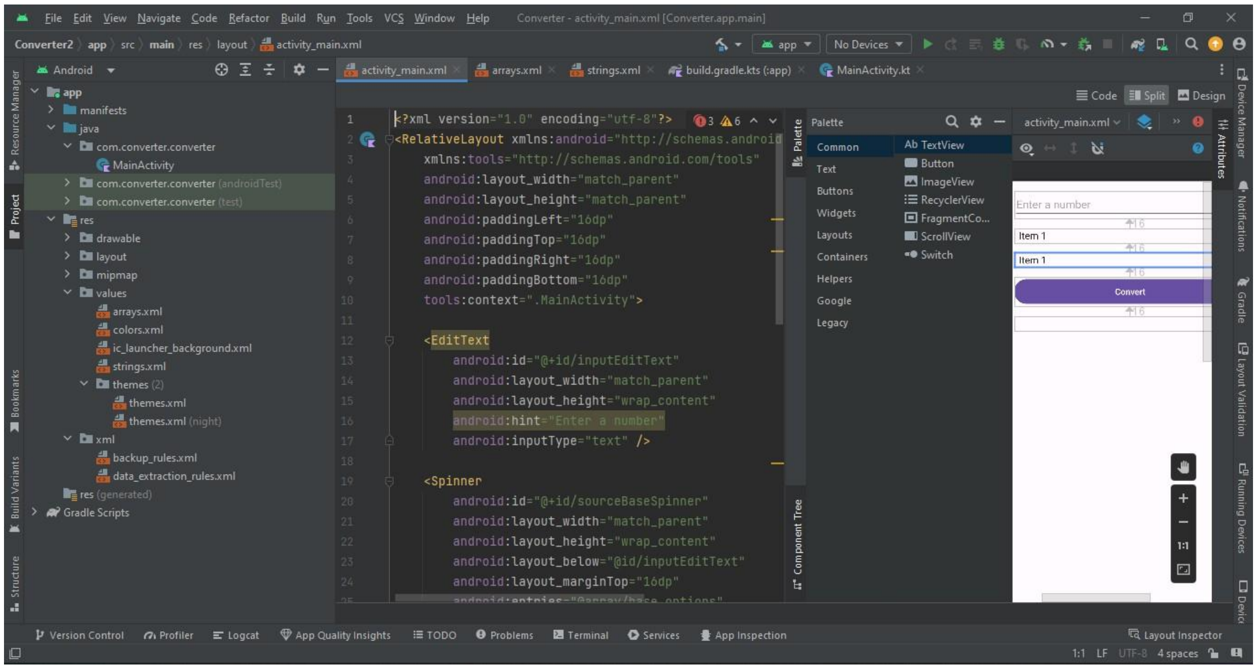


```
Enter a number: 1000100.10000
Enter the source base (2 for binary, 8 for octal, 10 for decimal, 16 for hexadecimal): 2
Enter the target base (2 for binary, 8 for octal, 10 for decimal, 16 for hexadecimal): 10
Result: 68.50000

--------------------------------
Process exited after 48.87 seconds with return value 0
Press any key to continue . . .
```

WE CONVERTED THE C++ CODE TO KOTLIN TO ENABLE IT TO RUN AS AN ANDROID APP



```kotlin
class MainActivity : AppCompatActivity() {
    private lateinit var inputEditText: EditText
    private lateinit var sourceBaseSpinner: Spinner
    private lateinit var targetBaseSpinner: Spinner
    private lateinit var resultTextView: TextView

    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setContentView(R.layout.activity_main)

        inputEditText = findViewById(R.id.inputEditText)
        sourceBaseSpinner = findViewById(R.id.sourceBaseSpinner)
        targetBaseSpinner = findViewById(R.id.targetBaseSpinner)
        resultTextView = findViewById(R.id.resultTextView)

        // Set up source base spinner
        val baseOptions = arrayOf("Binary (2)", "Octal (8)", "Decimal (10)", "Hexadecimal (16)")
        val adapter = ArrayAdapter(this, android.R.layout.simple_spinner_item, baseOptions)
        adapter.setDropDownViewResource(android.R.layout.simple_spinner_dropdown_item)
        sourceBaseSpinner.adapter = adapter

        // Set up target base spinner
        targetBaseSpinner.adapter = adapter

        // Set up button click listener
        findViewById<View>(R.id.convertButton).setOnClickListener { convertButtonClicked() }
```



```xml
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:paddingLeft="16dp"
    android:paddingTop="16dp"
    android:paddingRight="16dp"
    android:paddingBottom="16dp"
    tools:context=".MainActivity">

    <EditText
        android:id="@+id/inputEditText"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:hint="Enter a number"
        android:inputType="text" />

    <Spinner
        android:id="@+id/sourceBaseSpinner"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_below="@id/inputEditText"
        android:layout_marginTop="16dp"
```

C++ CODE CONVERTED TO KOTLIN

package com.converter.converter

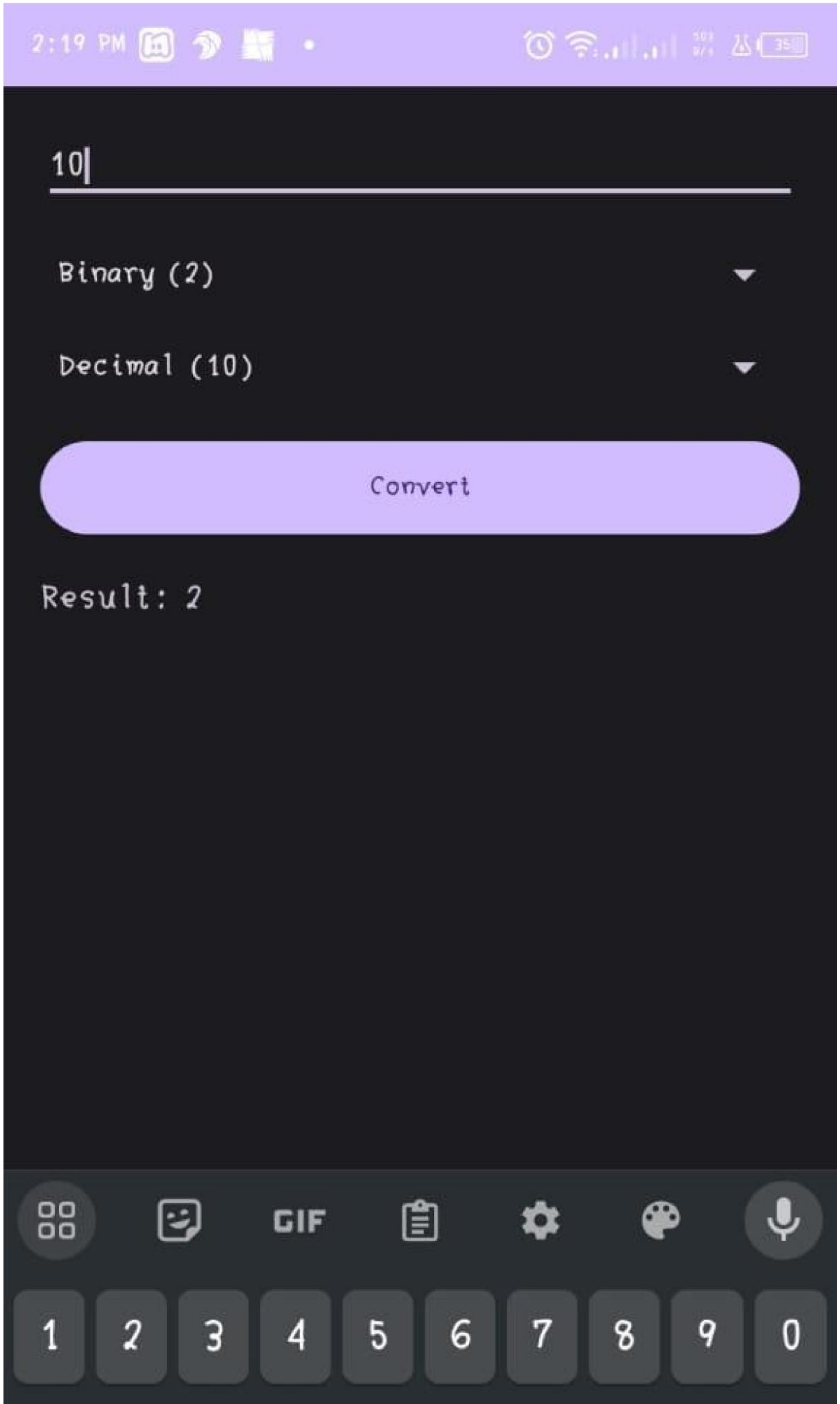
```
import android.os.Bundle import android.view.View

import android.widget.ArrayAdapter import

android.widget.EditText import

android.widget.Spinner import

android.widget.TextView import

androidx.appcompat.app.AppCompatActivity import

kotlin.math.pow

class MainActivity : AppCompatActivity() {

private lateinit var inputEditText: EditText

private lateinit var sourceBaseSpinner: Spinner

private lateinit var targetBaseSpinner: Spinner

private lateinit var resultTextView: TextView


   override fun onCreate(savedInstanceState: Bundle?) {

super.onCreate(savedInstanceState)

setContentView(R.layout.activity_main)


       inputEditText = findViewById(R.id.inputEditText)

sourceBaseSpinner = findViewById(R.id.sourceBaseSpinner)

targetBaseSpinner = findViewById(R.id.targetBaseSpinner)

resultTextView = findViewById(R.id.resultTextView)
```

```
    // Set up source base spinner        val baseOptions = arrayOf("Binary (2)", "Octal (8)",
"Decimal (10)", "Hexadecimal (16)")        val adapter = ArrayAdapter(this,
android.R.layout.simple_spinner_item, baseOptions)
adapter.setDropDownViewResource(android.R.layout.simple_spinner_dropdown_item)
sourceBaseSpinner.adapter = adapter

    // Set up target base spinner
targetBaseSpinner.adapter = adapter        // Set up button
click listener
findViewById<View>(R.id.convertButton).setOnClickListener
{ convertButtonClicked() }
  }

  private fun convertButtonClicked() {        val inputNum =
inputEditText.text.toString()        val sourceBase =
getBaseFromSpinner(sourceBaseSpinner)        val targetBase =
getBaseFromSpinner(targetBaseSpinner)

    val num = baseToDecimal(inputNum, sourceBase)
val result = decimalToBase(num, targetBase)

    resultTextView.text = "Result: $result"
  }
```

```
    private fun getBaseFromSpinner(spinner: Spinner): Int {
val selectedItem = spinner.selectedItem as String
return when {            selectedItem.contains("Binary") -> 2
selectedItem.contains("Octal") -> 8
selectedItem.contains("Decimal") -> 10
selectedItem.contains("Hexadecimal") -> 16
            else -> 10
        }
    }

    // Function to convert a decimal number to another base
private fun decimalToBase(num: Double, base: Int): String {
        var result = ""         var
intPart = num.toInt()        var
fracPart = num - intPart

        while (intPart > 0) {            val
remainder = intPart % base
result = if (remainder < 10) {
                ('0' + remainder).toString() + result
            } else {
                ('A' + remainder - 10).toString() + result
            }
            intPart /= base
        }


        if (fracPart > 0) {             result += "."             for (i in 0 until 5) {  // Specify
    the number of decimal places to convert                fracPart *= base
    val digit = fracPart.toInt()                result += if (digit < 10) {

                ('0' + digit).toString()
            } else {
                ('A' + digit - 10).toString()
            }
            fracPart -= digit
          }
        }


        return result
      }


      // Function to convert a number from any base to decimal
    private fun baseToDecimal(num: String, base: Int): Double {
        var result = 0.0        var intPart = 0
    var francPart = 0.0        val
    decimalPosition = num.indexOf('.')
```

```
        for (i in num.indices) {             if (i <
decimalPosition) {                  intPart = intPart * base +
if (num[i].isDigit()) {
                    (num[i] - '0')
                } else {
                    (num[i].uppercaseChar() - 'A' + 10)
                }
            } else if (i > decimalPosition) {
francPart += if (num[i].isDigit()) {
                    (num[i] - '0')
                } else {
                    (num[i].uppercaseChar() - 'A' + 10)
                } * base.toDouble().pow(i - decimalPosition)
            }
        }

        result = intPart + francPart
return result
    }
```

APP INTERFACE



10

Binary (2)

Decimal (10)

Result: 2