#include <iostream>

#include <cmath>

#include <string>

#include <sstream>

#include <algorithm>

#include <iomanip>

using namespace std;

int charToValue(char c) {

if (c >= '0' && c <= '9') return c - '0';

if (c >= 'A' && c <= 'F') return c - 'A' + 10;

if (c >= 'a' && c <= 'f') return c - 'a' + 10;

return -1;

}

char valueToChar(int v) {

if (v >= 0 && v <= 9) return v + '0';

if (v >= 10 && v <= 15) return v - 10 + 'A';

return '?';

}

double convertToDecimal(const string &num, int base) {

double result = 0;

size\_t pointPos = num.find('.');

for (size\_t i = 0; i < num.size(); ++i) {

if (num[i] == '.') continue;

int value = charToValue(num[i]);

if (value == -1 || value >= base) {

cerr << "Invalid number for base " << base << ": " << num << endl;

exit(1);

**}**

if (i < pointPos) {

result = result \* base + value;

} else {

result += value \* pow(base, -(int)(i - pointPos));

}

}

return result;

}

string convertFromDecimal(double num, int base) {

stringstream ss;

long long intPart = static\_cast<long long>(num);

double fracPart = num - intPart;

string intStr;

do {

intStr += valueToChar(intPart % base);

intPart /= base;

} while (intPart > 0);

reverse(intStr.begin(), intStr.end());

ss << intStr;

if (fracPart > 0) {

ss << '.';

int count = 0;

while (fracPart > 0 && count < 10) { // limit precision to 10 places

fracPart \*= base;

int digit = static\_cast<int>(fracPart);

ss << valueToChar(digit);

fracPart -= digit;

count++;

}

}

return ss.str();

}

int main() {

string num;

int fromBase, toBase;

cout << "Enter the number: ";

cin >> num;

cout << "Enter the base of the number: ";

cin >> fromBase;

cout << "Enter the base to convert to: ";

cin >> toBase;

double decimal = convertToDecimal(num, fromBase);

string result = convertFromDecimal(decimal, toBase);

cout << "The number " << num << " in base " << fromBase << " is " << result << " in base " << toBase << endl;

return 0;

}