// Q1. Write a C program for calculating the price of a product after adding the sales tax to its original price. Where the rate of tax and price is inputted by the user.

```
#include <stdio.h>
int main()
{
    float originalPrice, taxRate, totalPrice;
    printf("Enter the original price: ");
    scanf("%f", &originalPrice);
    printf("Enter the tax rate (in percentage): ");
    scanf("%f", &taxRate);
    totalPrice = originalPrice + (originalPrice * taxRate / 100);
    printf("Total price after tax: %.2f\n", totalPrice);
    return 0;
}
```

// Q2. Write a C program to calculate the weekly wages of an employee. The pay depends on wages per hour and the number of hours worked. Moreover, if the employee has worked for more than 30 hours, then he or she gets twice the wages per hour for every extra hour worked.

```
#include <stdio.h>
int main()
{
    float hourlyWage, weeklyWages;
    int hoursWorked;
    printf("Enter hourly wage: ");
    scanf("%f", &hourlyWage);
    printf("Enter hours worked: ");
    scanf("%d", &hoursWorked);
    if (hoursWorked > 30)
    {
        weeklyWages = (30 * hourlyWage) + ((hoursWorked - 30) * (2 * hourlyWage));
    }
    else
    {
        weeklyWages = hoursWorked * hourlyWage;
    }
    printf("Weekly wages: %.2f\n", weeklyWages);
    return 0;
}
```

// Q3. Mr. X goes to the market for buying some fruits and vegetables. Write a C program to find out the amount the shopkeeper will return to X after making a purchase.

```
#include <stdio.h>
int main()
{
    float currency = 500.0;
    float applePricePerKg = 50.0;
    float mangoPricePerKg = 35.0;
    float potatoPricePerKg = 10.0;
    float tomatoPricePerKg = 15.0;
    float totalCost = (2.0 * applePricePerKg) + (1.5 * mangoPricePerKg) + (2.5 * potatoPricePerKg) + (1.0 *
tomatoPricePerKg);
    float amountReturned = currency - totalCost;
    printf("Amount shopkeeper will return to X: %.2f\n", amountReturned);
    return 0;
}
```

// Q4. Write a C program to print your name, date of birth, and mobile number in 3 different lines.

```
#include <stdio.h>
int main()
{
    printf("Name: Nikhil Chauhan\n");
    printf("Date of Birth: 29-02-2004\n");
    printf("Mobile Number: 3636491069\n");
```

```
    return 0;
}
// Q5. Write a program to read an integer, a character, and a float value from the keyboard and display them on different lines on the screen.
#include <stdio.h>
int main()
{
    int integerNum;
    char character;
    float floatNum;
    printf("Enter an integer: ");
    scanf("%d", &integerNum);
    printf("Enter a character: ");
    scanf(" %c", &character);
    printf("Enter a float: ");
    scanf("%f", &floatNum);
    printf("Integer: %d\n", integerNum);
    printf("Character: %c\n", character);
    printf("Float: %.2f\n", floatNum);
    return 0;
}
// Q6. Write a program to print the following line (Assume the total value is contained in a variable named cost): "The sales total is: $172.53"
#include <stdio.h>
int main()
{
    float cost = 172.53;
    printf("The sales total is: $%.2f\n", cost);
    return 0;
}
// Q7. Raju got 6 and a half apples from each of Raghu, Sheenu, and Akash. Write a program that could help Raju find out how many apples he has in total without adding them.

#include <stdio.h>
int main()
{
    int rajuApples = 6;
    float halfApple = 0.5;
    float totalApples = rajuApples + 3 * halfApple;
    printf("Raju has %.1f apples in total.\n", totalApples);
    return 0;
}
// Q8. Write a program that prints the floating-point value in exponential format, correct to two decimal places.

#include <stdio.h>
int main()
{
    float floatValue = 12345.6789;
    printf("%.2e\n", floatValue);
    return 0;
}
// Q9. Write a program to input and print your mobile number (i.e., 10 digits).

#include <stdio.h>
int main()
{
    long long int mobileNumber;
    printf("Enter your 10-digit mobile number: ");
    scanf("%lld", &mobileNumber);
    printf("Mobile Number: %lld\n", mobileNumber);
    return 0;
}
```

// Q10. The population of a city is 30000. It increases by 20% during the first year and 30% during the second year. Write a program to find the population after two years.

```
#include <stdio.h>
int main()
{
    int initialPopulation = 30000;
    int populationAfterFirstYear = initialPopulation + (0.20 * initialPopulation);
    int populationAfterSecondYear = populationAfterFirstYear + (0.30 * populationAfterFirstYear);
    printf("Population after two years: %d\n", populationAfterSecondYear);
    return 0;
}
```

// Q11. Write a program to find the ASCII value of a character.

```
#include <stdio.h>
int main()
{
    char character;
    printf("Enter a character: ");
    scanf(" %c", &character);
    printf("ASCII value of %c is %d\n", character, character);
    return 0;
}
```

// Q12. Write a program to calculate the salary of an employee, given his basic pay (entered by the user), HRA (15% of the basic pay), and TA (20% of the basic pay).

```
#include <stdio.h>
int main()
{
    float basicPay, HRA, TA, totalSalary;
    printf("Enter the basic pay: ");
    scanf("%f", &basicPay);
    HRA = 0.15 * basicPay;
    TA = 0.20 * basicPay;

    // Calculate total salary
    totalSalary = basicPay + HRA + TA;

    // Display the total salary
    printf("Total Salary: %.2f\n", totalSalary);

    return 0;
}
```

// Q13. Write a program to find the slope of a line and the angle of inclination that passes through two points P and Q with coordinates (xp, yp) and (xq, yq) respectively.

```
#include <stdio.h>
#include <math.h>

int main()
{
    float xp, yp, xq, yq, slope, angle;
    printf("Enter the coordinates of point P (xp yp): ");
    scanf("%f %f", &xp, &yp);
    printf("Enter the coordinates of point Q (xq yq): ");
    scanf("%f %f", &xq, &yq);
    slope = (yq - yp) / (xq - xp);
    angle = atan(slope);
    printf("Slope: %.2f\n", slope);
    printf("Angle of Inclination (in degrees): %.2f\n", angle * (180.0 / M_PI));
    return 0;
}
```

// Q14. Write a program in C to calculate SPI (Semester Performance Index) for k = 5.

```c
#include <stdio.h>

int main()
{
    int k = 5;
    float gradePoints[] = {4.0, 3.5, 3.0, 2.5, 4.0};
    int credits[] = {3, 4, 3, 2, 4};
    float spi = 0.0;
    float totalGradePoints = 0.0;
    int totalCredits = 0;
    for (int i = 0; i < k; I++)
    {
        totalGradePoints += gradePoints[i] * credits[i];
        totalCredits += credits[i];
    }
    spi = totalGradePoints / totalCredits;
    printf("SPI for k = 5: %.2f\n", spi);
    return 0;
}
```

// Q15. Write a program to calculate the frequency (f) of a given wave with wavelength (λ) and speed ©, where c = λ * f.

```c
#include <stdio.h>
int main()
{
    float wavelength, speed, frequency;
    printf("Enter the wavelength (in meters): ");
    scanf("%f", &wavelength);
    printf("Enter the speed (in meters per second): ");
    scanf("%f", &speed);
    frequency = speed / wavelength;
    printf("Frequency: %.2f Hz\n", frequency);
    return 0;
}
```

// Q16. A car traveling at 30 m/s accelerates steadily at 5 m/s^2 for a distance of 70 m. What is the final velocity of the car? (Hint: v^2 = u^2 + 2as)

```c
#include <stdio.h>
#include <math.h>

int main()
{
    float initialVelocity = 30.0;
    float acceleration = 5.0;
    float distance = 70.0;
    float finalVelocity;
    finalVelocity = sqrt(pow(initialVelocity, 2) + 2 * acceleration * distance);
    printf("Final velocity: %.2f m/s\n", finalVelocity);
    return 0;
}
```

// Q17. A horse accelerates steadily from rest at 4 m/s^2 for 3 seconds. (a) What is its final velocity? (b) How far has it traveled? (Hint: (a) v = u + at, (b) s = ut + 0.5at^2)

```c
#include <stdio.h>

int main()
{
    float initialVelocity = 0.0;
    float acceleration = 4.0;
    float time = 3.0;
    float finalVelocity, distance;
```

```
    finalVelocity = initialVelocity + acceleration * time;
    distance = (initialVelocity * time) + (0.5 * acceleration * time * time);
    printf("(a) Final velocity: %.2f m/s\n", finalVelocity);
    printf("(b) Distance traveled: %.2f meters\n", distance);
    return 0;
}
// Q18. Write a program to find the sum of the last four digits of your university roll number.

#include <stdio.h>
int main()
{
    int rollNumber = 12345678;
    int lastFourDigits = rollNumber % 10000;
    int sum = 0;
    while (lastFourDigits > 0)
    {
        sum += lastFourDigits % 10;
        lastFourDigits /= 10;
    }
    printf("Sum of the last four digits of the roll number: %d\n", sum);
    return 0;
}
// Q19. Write a program to initialize your height and weight in cm and kg, respectively, demonstrating compile-time initialization, and convert them into feet and pounds, respectively. Note: 1 cm = 0.393701 inch, 1 kg = 2.20462 pounds.

#include <stdio.h>

int main()
{
    float heightCm = 175.0;
    float weightKg = 70.0;
    float heightFeet = heightCm * 0.393701 / 12.0; // 1 inch = 1/12 feet
    float weightPounds = weightKg * 2.20462;
    printf("Height in feet: %.2f feet\n", heightFeet);
    printf("Weight in pounds: %.2f pounds\n", weightPounds);
    return 0;
}
// Q20. Code the variable declarations for each of the following:
// A character variable named option.
// An integer variable sum initialized to 0.
// A floating-point variable, product, initialized to 1.
#include <stdio.h>
int main()
{
    char option;
    int sum = 0;
    float product = 1.0;
    return 0;
}
// Q21. Write a program that reads nine integers and displays these numbers by printing three numbers in a line separated by commas.

#include <stdio.h>
int main()
{
    int numbers[9];
    printf("Enter nine integers:\n");
    for (int i = 0; i < 9; I++)
    {
        scanf("%d", &numbers[i]);
    }
```

```
    for (int i = 0; i < 9; i += 3)
    {
        printf("%d, %d, %d\n", numbers[i], numbers[i + 1], numbers[i + 2]);
    }
    return 0;
}
```
// Q22. What are header files, and what are their uses in C programming?

/*Header files in C are files that contain declarations and prototypes for functions and data structures. They are used to declare functions, macros, constants, and data types that can be used across multiple source code files. The primary uses of header files in C programming are:

Encapsulation: Header files encapsulate the interfaces for libraries or modules, allowing you to hide the implementation details from the user of the library.

Code Reusability: They enable code reuse by allowing multiple source files to include the same set of function declarations, ensuring consistent usage.

Error Prevention: Header files help catch errors at compile-time by ensuring that function calls and data structures are used correctly.

Modularity: Header files promote modularity in your code by separating different components or modules into separate files.

Documentation: Header files serve as documentation by providing information about the functions and data structures available in a module.

Examples of commonly used header files in C include <stdio.h> for standard input/output functions, <stdlib.h> for standard library functions, and <math.h> for mathematical functions.
*/
// Q23. What will be the output of the following program?

```
#include <stdio.h>
int main()
{
    int num = 070;
    printf("%d\t%o\t%x", num, num, num);
    return 0;
}
```
// Explanation: In the program, num is assigned the octal value 070, which is equivalent to the decimal value 56. The printf statement prints num in decimal, octal, and hexadecimal formats.
// Q24. What will be the output of the following program?

```
#include <stdio.h>
void main()
{
    int x = printf("GLA UNIVERSITY");
    printf("%d", x);
}
```
/*Explanation: In this program, x is assigned the return value of the printf function, which is the number of characters printed. The printf function inside main prints "GLA UNIVERSITY," which contains 14 characters. So, x will be assigned the value 14. The second printf statement will print the value of x, which is 14.*/

// Q25. What are library functions, and list any four library functions.

/*Library functions in C are pre-written functions provided by the C standard library and other libraries that can be used to perform various tasks without having to write the code for those tasks from scratch. These functions are included in header files and can be used by including the respective header files in your C programs. Here are four commonly used library functions in C:

printf: This function is used for formatted output and is part of the standard I/O library (<stdio.h>). It allows you to display text and data with specified formatting.

scanf: This function is used for formatted input and is also part of the standard I/O library (<stdio.h>). It allows you to read data from the user or a file with specified formatting.

strlen: This function is used to determine the length of a null-terminated string and is part of the C Standard Library (<string.h>).

rand: This function generates a pseudo-random integer and is part of the C Standard Library (<stdlib.h>). It is often used for tasks involving randomness.
*/
// Q26. What will be the output of the following program?
```
#include <stdio.h>
void main()
{
    int x = printf("C is a placement-oriented Language") - printf("Hi");
    printf("%d %o %x", x, x, x);
}
```
/*Explanation: In this program, x is assigned the result of the subtraction between the return value of the first printf function (which prints "C is a placement-oriented Language") and the return value of the second printf function (which prints "Hi"). Both printf calls return the number of characters printed.

The first printf prints 34 characters.
The second printf prints 2 characters.
So, x will be assigned the value 34 - 2 = 32. The second printf statement will print the value of x in decimal, octal, and hexadecimal formats.
*/
// Q27. What is the meaning of the following statement? printf("%d", scanf("%d%d", &a, &b));

/*The statement printf("%d", scanf("%d%d", &a, &b)); is a bit unusual. It combines the scanf and printf functions in a single statement. Here's how it works:

scanf("%d%d", &a, &b);: This part of the statement reads two integers from the user input and stores them in variables a and b. It uses the format specifier %d twice, expecting two integer inputs separated by whitespace.

printf("%d", ...);: This part of the statement is the printf function that prints the result. However, the result of scanf is the number of successfully read items. In this case, if scanf successfully reads two integers, it will return 2.

So, the statement effectively reads two integers into a and b and then prints 2 using printf.
*/
// Q28. What will be the output of the following program?

```
#include <stdio.h>
void main()
{
    printf(" \"C %% FOR %% PLACEMENT\"");
}
```
/*Explanation: In this program, the printf statement is used to print the string "C % FOR % PLACEMENT". To include a literal percent sign in the output, you need to use a double percent sign %%. Therefore, the output will be:
 */

// Q29. Suppose the distance between GLA University and Delhi is m kilometers (to be entered by the user). By bus, it takes 8 hours to reach Delhi from GLA University, and by train, it takes 10 hours to reach. Write a program to calculate the speed of the bus and train in km/hr.

```
#include <stdio.h>
int main()
{
    float distance;
    float busSpeed, trainSpeed;
    printf("Enter the distance between GLA University and Delhi (in kilometers): ");
    scanf("%f", &distance);
```

```
    busSpeed = distance / 8.0;
    trainSpeed = distance / 10.0;
    printf("Speed of bus: %.2f km/hr\n", busSpeed);
    printf("Speed of train: %.2f km/hr\n", trainSpeed);
    return 0;
}

// Q29. Develop a 'C' program to calculate the speed of a bus given that it takes 4 hours to travel a certain distance.
#include <stdio.h>
int main()
{
    float distance, time, speed;
    printf("Enter the distance (in kilometers): ");
    scanf("%f", &distance);
    printf("Enter the time taken (in hours): ");
    scanf("%f", &time);
    speed = distance / time;
    printf("The speed of the bus is %.2f km/h\n", speed);
    return 0;
}
// Q30. Write a 'C' program to find the average marks of three participants who scored 50, 70, and 80 marks.

#include <stdio.h>
int main()
{
    int satyamMarks = 50, sumanMarks = 70, shyamMarks = 80;
    float average;
    average = (satyamMarks + sumanMarks + shyamMarks) / 3.0;
    printf("The average marks of the three participants are: %.2f\n", average);
    return 0;
}
// Q31. Develop a 'C' program to help Mohan rectify his mistake of giving money to Saurav and Sajal in the wrong order.

#include <stdio.h>
int main()
{
    int sauravMoney, sajalMoney;
    printf("Enter the amount given to Saurav: ");
    scanf("%d", &sauravMoney);
    printf("Enter the amount given to Sajal: ");
    scanf("%d", &sajalMoney);
    int temp = sauravMoney;
    sauravMoney = sajalMoney;
    sajalMoney = temp;
    printf("Mohan should give %d to Saurav and %d to Sajal.\n", sauravMoney, sajalMoney);
    return 0;
}
// Q32. Develop a 'C' program to calculate the distance traveled by you when running at a speed of 4 km/h for 3 minutes.

#include <stdio.h>
int main()
{
    float speed = 4.0;
    float time = 3.0 / 60.0;
    float distance;
    distance = speed * time;

    printf("The distance traveled is %.2f km\n", distance);
```

```
    return 0;
}
// Q33. Can two or more escape sequences such as \n and \t be combined in a single line of program code?

/*Yes, you can combine multiple escape sequences in a single line of program code. For example, you can have a string with both newline (\n) and tab (\t) characters within it. Here's an example:
 */
#include <stdio.h>
int main()
{
    printf("This is a line with a newline character (\n) and a tab character (\t).\n");
    return 0;
}
// Q34. What are comments, and how do you insert them in a C program?

/*Comments in C are used to provide explanations or descriptions within your code. They are ignored by the compiler and serve as documentation for programmers. In C, there are two types of comments:

Single-line comments: These comments start with // and continue until the end of the line. They are used for short comments on a single line.

// This is a single-line comment
Multi-line comments: These comments start with /* and end with */
// They can span multiple lines and are used for longer explanations.

/*
   This is a multi-line comment.
   It can span multiple lines.
*/
// Comments are used to make your code more understandable and maintainable. They help you and other programmers understand the purpose of code segments.

// Q35. What is wrong in this statement? scanf("%d", number);

// In the statement scanf("%d", number);
// We will have to use & before the variable
// scanf("%d", &number);
// The & operator is used to get the address of the number variable for scanf to store the input value correctly.

// Q36. What will be the output of the following program?

#include <stdio.h>
int main()
{
    if (sizeof(int) > -1)
        printf("Yes");
    else
        printf("No");
    return 0;
}
/*Explanation: In C, the sizeof operator returns the size (in bytes) of a data type or an expression. In most systems, the sizeof(int) will be greater than -1 because int is typically represented using more than one byte. Therefore, the condition in the if statement will be true, and "Yes" will be printed.*/

// Q37. Point out which of the following variable names are invalid: gross-salary, INTEREST, salary of emp, avg., thereisbookinmysoup

/*Here are the points for each variable name:

gross-salary: Valid (Variable names can contain letters, numbers, and underscores, but cannot start with a number.)
```

INTEREST: Valid (Variable names are case-sensitive, and uppercase letters are different from lowercase letters.)

salary of emp: Invalid (Variable names cannot contain spaces. You can use underscores to separate words, like salary_of_emp.)

avg.: Invalid (Variable names cannot contain special characters like .. You can use underscores, like avg_score.)

thereisbookinmysoup: Valid (This variable name contains only letters, and underscores, which are allowed.)
*/
// Q38. Develop a 'C' program to help Tom calculate the time required to completely clean a 175-gallon reef tank.

```
#include <stdio.h>

int main()
{
    float rate = 25.0;      // Rate of draining in gallons per hour
    float tankSize = 175.0; // Size of the tank in gallons
    float time;

    // Calculate the time required (tank size divided by draining rate)
    time = tankSize / rate;

    // Display the time in hours
    printf("Tom will take %.2f hours to completely clean the tank.\n", time);

    return 0;
}
```

// Q39. Develop a 'C' program to calculate after how many hours the battery power is at 75% given the formula y = -0.2x + 1, where y is the battery power (in decimal form) and x is the time (in hours).

```
#include <stdio.h>

int main()
{
    float batteryPower = 0.75; // 75% battery power
    float time;

    // Solve for time using the formula: y = -0.2x + 1
    time = (1 - batteryPower) / 0.2;

    // Display the time in hours
    printf("The battery power will be at 75%% after %.2f hours.\n", time);

    return 0;
}
```

// Q40. Which of the following is used to convert the high-level language into machine language in a single go?
// a. Compiler b. Interpreter c. Linker d. Assembler

// Answer: a. Compiler

// Q41. What is the format specifier for an Octal Number?
// a. %0 b. %d c. %o d. %e

// Answer: c. %o

// Q42. Which format specifier is used to print the exponent value up to 2 decimal places?
// a. %e b. %.2f c. %f d. %.2e

// Answer: d. %.2e

```
// Q43. Which of the following is not a basic data type?
// a. char b. array c. float d. int

// Answer: b. array

// Q44. What is the output of the following code?

#include <stdio.h>

int main()
{
    int x = 0;
    x = printf("\"hello\\b\"");
    printf("%d", x);
}
// Answer: c. "hell"8

// Q45. What is the output of the following code?

#include <stdio.h>

int main()
{
    int b, c = 5;
    printf(" %d , %d", b, c);
}
// Answer: c. Garbage, 5.000000

// Q46. Which of the following is an identifier?
// a. &fact b. Basic_pay c. enum d. 1sum

// Answer: b. Basic_pay

// Q47. What is the output of the following program?

#include <stdio.h>
int main()
{
    char x, a = 'c';
    x = printf("%c", a);
    printf("%d", x);
}
// Answer: a. c1

// Q48. Perform the following conversion from Decimal to other number systems as directed:
// a) (365.55)10 = (?)2

#include <stdio.h>
int main()
{
    int decimal = 365;
    printf("Binary: ");
    for (int i = 31; i >= 0; I--)
    {
        int bit = (decimal >> i) & 1;
        printf("%d", bit);
    }
    printf("."); // Decimal point
    decimal = 55;
    for (int i = 0; i < 8; I++)
    {
```

```
        decimal *= 2;
        int bit = decimal / 100;
        printf("%d", bit);
        decimal %= 100;
    }
    printf("\n");
    return 0;
}
// b) (453.65)10 = (?)8

// To convert a decimal number to octal, you can use the following C code:
#include <stdio.h>
int main()
{
    int decimal = 453;
    printf("Octal: ");
    while (decimal > 0)
    {
        int octalDigit = decimal % 8;
        printf("%d", octalDigit);
        decimal /= 8;
    }
    printf(".");
    decimal = 65;
    for (int i = 0; i < 4; I++)
    {
        decimal *= 8;
        int octalDigit = decimal / 100;
        printf("%d", octalDigit);
        decimal %= 100;
    }
    printf("\n");
    return 0;
}
// c) (5164.12)10 = (?)16

// To convert a decimal number to hexadecimal, you can use the following C code:
#include <stdio.h>
int main()
{
    int decimal = 5164;
    printf("Hexadecimal: ");
    while (decimal > 0)
    {
        int hexDigit = decimal % 16;
        if (hexDigit < 10)
        {
            printf("%d", hexDigit);
        }
        else
        {
            printf("%c", 'A' + (hexDigit - 10));
        }
        decimal /= 16;
    }
    printf(".");
    decimal = 12;
    for (int i = 0; i < 3; I++)
    {
        decimal *= 16;
        int hexDigit = decimal / 256;
        if (hexDigit < 10)
```

```
        {
            printf("%d", hexDigit);
        }
        else
        {
            printf("%c", 'A' + (hexDigit - 10));
        }
        decimal %= 256;
    }
    printf("\n");
    return 0;
}

// d) (23.65)10 = (?)5

// To convert a decimal number to a base-5 (quinary) representation, you can use the following C code:
#include <stdio.h>

int main()
{
    int decimal = 23;
    printf("Base-5: ");
    while (decimal > 0)
    {
        int base5Digit = decimal % 5;
        printf("%d", base5Digit);
        decimal /= 5;
    }
    printf("."); // Decimal point
    decimal = 65;
    for (int i = 0; i < 3; I++)
    {
        decimal *= 5;
        int base5Digit = decimal / 100;
        printf("%d", base5Digit);
        decimal %= 100;
    }
    printf("\n");
    return 0;
}

// e) (772)10 = (?)7

// To convert a decimal number to base-7, you can use the following C code:
#include <stdio.h>

int main()
{
    int decimal = 772;
    printf("Base-7: ");
    while (decimal > 0)
    {
        int base7Digit = decimal % 7;
        printf("%d", base7Digit);
        decimal /= 7;
    }
    printf("\n");
    return 0;
}
// Q49. Convert the following numbers to the decimal number system:

// a) (325.54)6 = (?)10
```

```c
#include <stdio.h>
#include <math.h>
int main()
{
    int base6Number = 32554;
    int decimalNumber = 0;
    int position = 0;
    while (base6Number > 0)
    {
        int digit = base6Number % 10;
        decimalNumber += digit * pow(6, position);
        base6Number /= 10;
        position++;
    }
    printf("Decimal: %d\n", decimalNumber);
    return 0;
}
// b) (1001010110101.1110101)2 = (?)10
// To convert a binary number to decimal, you can use the following C code:

#include <stdio.h>
#include <math.h>
int main()
{
    long long binaryNumber = 1001010110101;
    double decimalNumber = 0;
    int position = 0;
    while (binaryNumber > 0)
    {
        int digit = binaryNumber % 10;
        decimalNumber += digit * pow(2, position);
        binaryNumber /= 10;
        position++;
    }
    printf("Decimal: %.5f\n", decimalNumber);
    return 0;
}
// This code converts the binary number 1001010110101.1110101 to decimal.

// c) (742.72)8 = (?)10
#include <stdio.h>
#include <math.h>
int main()
{
    int octalNumber = 74272;
    int decimalNumber = 0;
    int position = 0;
    while (octalNumber > 0)
    {
        int digit = octalNumber % 10;
        decimalNumber += digit * pow(8, position);
        octalNumber /= 10;
        position++;
    }
    printf("Decimal: %d\n", decimalNumber);
    return 0;
}
// d) (AC94.C5)16 = (?)10
// To convert a hexadecimal number to decimal, you can use the following C code:
#include <stdio.h>
#include <math.h>
```

```
int main()
{
    char hexNumber[] = "AC94.C5";
    double decimalNumber = 0;
    int position = 0;
    for (int i = 0; hexNumber[i] != '\0'; i++)
    {
        if (hexNumber[i] >= '0' && hexNumber[i] <= '9')
        {
            decimalNumber = decimalNumber * 16 + (hexNumber[i] - '0');
        }
        else if (hexNumber[i] >= 'A' && hexNumber[i] <= 'F')
        {
            decimalNumber = decimalNumber * 16 + (hexNumber[i] - 'A' + 10);
        }
        else if (hexNumber[i] >= 'a' && hexNumber[i] <= 'f')
        {
            decimalNumber = decimalNumber * 16 + (hexNumber[i] - 'a' + 10);
        }
        if (hexNumber[i] == '.')
        {
            position = -1;
        }
        else if (position >= 0)
        {
            position++;
        }
    }
    printf("Decimal: %.5f\n", decimalNumber / pow(16, position));
    return 0;
}
// Q50. Perform the following conversion from Hexadecimal to other number systems as directed:

// (DB56.CD4)16 = (?)2, (?)8, (?)4
#include <stdio.h>
#include <string.h>
char *hexToBinary(char hexDigit)
{
    switch (hexDigit)
    {
    case '0':
        return "0000";
    case '1':
        return "0001";
    case '2':
        return "0010";
    case '3':
        return "0011";
    case '4':
        return "0100";
    case '5':
        return "0101";
    case '6':
        return "0110";
    case '7':
        return "0111";
    case '8':
        return "1000";
    case '9':
        return "1001";
    case 'A':
        return "1010";
```

```
    case 'B':
        return "1011";
    case 'C':
        return "1100";
    case 'D':
        return "1101";
    case 'E':
        return "1110";
    case 'F':
        return "1111";
    default:
        return "";
    }
}

char *binaryToOctal(char *binary)
{
    static char octal[12];
    memset(octal, '0', sizeof(octal));
    int len = strlen(binary);
    int octalIndex = 0;
    int carry = 0;
    int padding = (3 - (len % 3)) % 3;
    for (int i = 0; i < padding; I++)
    {
        binary[len + i] = '0';
    }
    for (int i = len + padding - 1; i >= 0; I--)
    {
        int bit = binary[i] - '0';
        int sum = carry + (bit << 2) + (bit << 1);
        octal[octalIndex++] = '0' + (sum % 10);
        carry = sum / 10;
    }
    if (carry > 0)
    {
        octal[octalIndex++] = '0' + carry;
    }
    int start = 0;
    int end = octalIndex - 1;
    while (start < end)
    {
        char temp = octal[start];
        octal[start] = octal[end];
        octal[end] = temp;
        start++;
        end--;
    }
    return octal;
}
char *binaryToQuaternary(char *binary)
{
    static char quaternary[16];
    memset(quaternary, '0', sizeof(quaternary));

    int len = strlen(binary);
    int quaternaryIndex = 0;
    int carry = 0;
    int padding = (2 - (len % 2)) % 2;
    for (int i = 0; i < padding; I++)
    {
        binary[len + i] = '0';
```

```
    }

    for (int i = len + padding - 1; i >= 0; I--)
    {
        int bit = binary[i] - '0';
        int sum = carry + (bit << 1);
        quaternary[quaternaryIndex++] = '0' + (sum % 4);
        carry = sum / 4;
    }

    if (carry > 0)
    {
        quaternary[quaternaryIndex++] = '0' + carry;
    }
    int start = 0;
    int end = quaternaryIndex - 1;
    while (start < end)
    {
        char temp = quaternary[start];
        quaternary[start] = quaternary[end];
        quaternary[end] = temp;
        start++;
        end--;
    }
    return quaternary;
}

int main()
{
    char hexNumber[] = "DB56.CD4";
    char binaryNumber[50] = "";
    char octalNumber[50] = "";
    char quaternaryNumber[50] = "";
    int binaryIndex = 0;
    for (int i = 0; hexNumber[i] != '\0'; i++)
    {
        if (hexNumber[i] != '.')
        {
            char *binaryDigit = hexToBinary(hexNumber[i]);
            strcpy(binaryNumber + binaryIndex, binaryDigit);
            binaryIndex += strlen(binaryDigit);
        }
        else
        {
            binaryNumber[binaryIndex++] = '.';
        }
    }
    strcpy(octalNumber, binaryToOctal(binaryNumber));
    strcpy(quaternaryNumber, binaryToQuaternary(binaryNumber));

    printf("(DB56.CD4)16 in binary: %s\n", binaryNumber);
    printf("(DB56.CD4)16 in octal: %s\n", octalNumber);
    printf("(DB56.CD4)16 in quaternary: %s\n", quaternaryNumber);

    return 0;
}
```