In [ ]:
```c
# prompt: Write a ternary operator statement to calculate the PWM duty cycle.
# cycle should be determined based on the following conditions:
# 1) If the temperature value (Temp) is greater than or equal to 50 and the in
# switch (Switch) is pressed (1), then:
# a) If the system mode (Mode) is automatic (1), set PWM_Duty to 75.
# b) Otherwise, set PWM_Duty to 50.
# 2) If the above conditions are not met, set PWM_Duty to 25.
# In C

#include <stdio.h>

int main() {
  int Temp = 60; // Example temperature value
  int Switch = 1; // Example switch state (pressed)
  int Mode = 1; // Example system mode (automatic)
  int PWM_Duty;

  PWM_Duty = (Temp >= 50 && Switch == 1) ? ((Mode == 1) ? 75 : 50) : 25;

  printf("PWM Duty Cycle: %d\n", PWM_Duty);
  return 0;
}
```

In [ ]:
```c
# prompt: A cashier has currency notes of denominations 10, 50 and 100. If the
# withdrawn is input through the keyboard in hundreds, find the total number c
# currency notes of each denomination the cashier will have to give to the
# withdrawer.

amount = int(input("Enter the amount to be withdrawn (in hundreds): "))

if amount < 0:
    print("Invalid amount")
else:
    num_100 = amount
    amount %= 100

    num_50 = amount // 50
    amount %= 50

    num_10 = amount // 10
    amount %= 10

    if amount != 0:
        print("Cannot dispense exact amount with given denominations.")
    else:
        print("Number of 100 notes:", num_100)
        print("Number of 50 notes:", num_50)
        print("Number of 10 notes:", num_10)
```

In [ ]:
```c
# prompt: Write a program to display all leap years between two given years
# (inclusive).Input the start year and end year from the user.Print all leap y
# that range, one per line. in C
```

```c
#include <stdio.h>

int main() {
    int start_year, end_year;

    printf("Enter the start year: ");
    scanf("%d", &start_year);

    printf("Enter the end year: ");
    scanf("%d", &end_year);

    printf("Leap years between %d and %d are:\n", start_year, end_year);

    for (int year = start_year; year <= end_year; year++) {
        if ((year % 4 == 0 && year % 100 != 0) || (year % 400 == 0)) {
            printf("%d\n", year);
        }
    }

    return 0;
}
```

In [ ]:
```c
// prompt: Write a C program that takes a string as input from the user. The pr
// then calculate and print the sum of the ASCII values of all the characters i
// string

#include <stdio.h>
#include <string.h>

int main() {
    char str[100];
    int sum = 0;

    printf("Enter a string: ");
    fgets(str, sizeof(str), stdin); // Use fgets to prevent buffer overflow

    // Remove the trailing newline character if present
    str[strcspn(str, "\n")] = 0;


    for (int i = 0; i < strlen(str); i++) {
        sum += (int)str[i];
    }

    printf("Sum of ASCII values: %d\n", sum);
    return 0;
}
```

In [ ]:
```c
// prompt: Maximum possible score in the particular game is 1000. Assume score
// user. Calculate the percantage of the user's score in relation to the maximu
// available score.
// Convert userScore to float to make sure that the division is accurate in C
```

```c
#include <stdio.h>

int main() {
    float userScore;
    float percentage;

    printf("Enter the user's score: ");
    scanf("%f", &userScore);

    // Calculate the percentage
    percentage = (userScore / 1000) * 100;

    printf("The user's score percentage is: %.2f%%\n", percentage);

    return 0;
}
```

In [ ]:
```c
# prompt: Write a C program to convert specified days into years, weeks and da
# Note: Ignore leap year.

#include <stdio.h>

int main() {
    int days, years, weeks;

    printf("Enter the number of days: ");
    scanf("%d", &days);

    years = days / 365;
    weeks = (days % 365) / 7;
    days = days - (years * 365) - (weeks * 7);

    printf("Years: %d\n", years);
    printf("Weeks: %d\n", weeks);
    printf("Days: %d\n", days);

    return 0;
}
```

In [ ]:
```c
# prompt: Write a C program to convert a binary number (input as a string) int
# decimal equivalent.

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

int main() {
    char binary[100];
    int decimal = 0, power = 0;

    printf("Enter a binary number: ");
    scanf("%s", binary);
```

```c
    int len = strlen(binary);
    for (int i = len - 1; i >= 0; i--) {
        if (binary[i] == '1') {
            decimal += pow(2, power);
        } else if (binary[i] != '0') {
            printf("Invalid binary number.\n");
            return 1; // Indicate an error
        }
        power++;
    }

    printf("Decimal equivalent: %d\n", decimal);
    return 0;
}
```

In [ ]:

```c
// prompt: Write a program in C that takes an integer input n
// (1 ≤ n ≤ 10,000) and performs the following operations:
// If n is divisible by both 3 and 5, print "RadhaKrushna".
// If n is divisible by 3 but not by 5, print "Radha".
// If n is divisible by 5 but not by 3, print "Krushna".

#include <stdio.h>

int main() {
    int n;

    printf("Enter an integer (1 ≤ n ≤ 10,000): ");
    scanf("%d", &n);

    if (n < 1 || n > 10000) {
        printf("Invalid input. Please enter a number between 1 and 10,000.\n")
        return 1; // Indicate an error
    }

    if (n % 3 == 0 && n % 5 == 0) {
        printf("RadhaKrushna\n");
    } else if (n % 3 == 0) {
        printf("Radha\n");
    } else if (n % 5 == 0) {
        printf("Krushna\n");
    } else {
        // Handle cases where n is not divisible by 3 or 5 (optional)
        // printf("Neither divisible by 3 nor 5\n");
    }

    return 0;
}
```

In [ ]:

```c
// prompt: check if n is a prime number:
// If it is, print "Prime".
// If not, print the sum of all digits in n that are odd.
// If there are no odd digits, print "No Odd Digits".
// Constraints: Use only nested if-else and avoid using external libraries for
```

```c
# In C

#include <stdio.h>
#include <stdbool.h>

// Function to check if a number is prime
bool is_prime(int n) {
    if (n <= 1) {
        return false;
    }
    if (n <= 3) {
        return true;
    }
    if (n % 2 == 0 || n % 3 == 0) {
        return false;
    }
    for (int i = 5; i * i <= n; i = i + 6) {
        if (n % i == 0 || n % (i + 2) == 0) {
            return false;
        }
    }
    return true;
}

int main() {
    int n;
    printf("Enter an integer: ");
    scanf("%d", &n);

    if (n == 2) {
        printf("Prime\n");
        return 0;
    }

    if (is_prime(n)) {
        printf("Prime\n");
    } else {
        int sum_of_odd_digits = 0;
        int temp = n;
        bool has_odd_digit = false;

        while (temp > 0) {
            int digit = temp % 10;
            if (digit % 2 != 0) {
                sum_of_odd_digits += digit;
                has_odd_digit = true;
            }
            temp /= 10;
        }

        if (has_odd_digit) {
            printf("%d\n", sum_of_odd_digits);
        } else {
```

```
            printf("No Odd Digits\n");
        }
    }

    return 0;
}
```

In [ ]:
```
# prompt: Write a program in cto check whether a number is a Krishnamurty numb
# of digits equals the number

#include <stdio.h>

// Function to calculate factorial of a number
int factorial(int n) {
    int fact = 1;
    for (int i = 1; i <= n; i++) {
        fact *= i;
    }
    return fact;
}

int main() {
    int num, originalNum, digit, sum = 0;

    printf("Enter a number: ");
    scanf("%d", &num);

    originalNum = num; // Store the original number

    while (num > 0) {
        digit = num % 10;
        sum += factorial(digit);
        num /= 10;
    }

    if (sum == originalNum) {
        printf("%d is a Krishnamurty number.\n", originalNum);
    } else {
        printf("%d is not a Krishnamurty number.\n", originalNum);
    }

    return 0;
}
```

In [ ]:
```
# prompt: Write a program to print all prime numbers from 1 to 300. (Hint: Use

#include <stdio.h>
#include <stdbool.h>

// Function to check if a number is prime
bool is_prime(int n) {
    if (n <= 1) {
        return false;
```

```c
    }
    for (int i = 2; i * i <= n; i++) {
        if (n % i == 0) {
            return false;
        }
    }
    return true;
}

int main() {
    printf("Prime numbers from 1 to 300:\n");
    for (int i = 2; i <= 300; i++) {
        if (is_prime(i)) {
            printf("%d ", i);
        }
    }
    printf("\n");
    return 0;
}
```

In [ ]:

```c
// prompt: Write a C program that accepts a sequence of different values and ca
// value.
// The sum of the values before the maximum value is 0, if there are no values
// after the maximum value is 0, if there are no values after the maximum. with

#include <stdio.h>
#include <limits.h>

int main() {
    int n, i, max_val, max_index;
    int sum_before = 0, sum_after = 0;

    printf("Enter the number of values: ");
    scanf("%d", &n);

    int values[n];

    printf("Enter the values:\n");
    for (i = 0; i < n; i++) {
        scanf("%d", &values[i]);
    }

    // Find the maximum value and its index
    max_val = INT_MIN;
    max_index = -1;
    for (i = 0; i < n; i++) {
        if (values[i] > max_val) {
            max_val = values[i];
            max_index = i;
        }
    }

    if (max_index == -1) {
```

```
        printf("No values entered.\n");
        return 1;
    }

    // Calculate the sum before the maximum value
    for (i = 0; i < max_index; i++) {
        sum_before += values[i];
    }

    // Calculate the sum after the maximum value
    for (i = max_index + 1; i < n; i++) {
        sum_after += values[i];
    }

    printf("Sum of values before the maximum value: %d\n", sum_before);
    printf("Sum of values after the maximum value: %d\n", sum_after);

    return 0;
}
```

In [ ]:
```
# prompt: rite a C program to print the following character number pyramid as:
# 1
# B B
# 2 3 4
# D D D D
# 5 6 7 8 9

#include <stdio.h>

int main() {
    int num = 1;
    char ch = 'A';

    for (int i = 1; i <= 5; i++) {
        for (int j = 1; j <= i; j++) {
            if (i % 2 != 0) {
                printf("%d ", num++);
            } else {
                printf("%c ", ch++);
            }
        }
        printf("\n");
        if (i % 2 == 0) {
            ch = ch - i + 1; // Reset the character for the next row if it's a
            ch = ch + 'A' - '0'; // Convert back to the appropriate character
        }
    }
    return 0;
}
```

In [ ]:
```
# prompt: Write a C program that asks the user to enter a password. If the ent
# statement to allow the user to try again up to 3 times before displaying an
```

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

int main() {
    char password[20];
    char correct_password[20] = "password123"; // Replace with your desired pa
    int attempts = 0;

    retry:
    printf("Enter the password: ");
    scanf("%s", password);

    if (strcmp(password, correct_password) == 0) {
        printf("Password correct!\n");
    } else {
        attempts++;
        if (attempts < 3) {
            printf("Incorrect password. Please try again.\n");
            goto retry;
        } else {
            printf("Too many incorrect attempts. Access denied.\n");
        }
    }

    return 0;
}
```