Lab Exercise 1: Demonstrate Nested For Loop

Objective:
To understand the use of nested `for` loops in C.

Problem Statement:
Write a C program using nested `for` loops to print a right-angled triangle pattern of stars (`*`).

Algorithm:

1. Start
2. Read the number of rows `n`
3. For each row `i` from 1 to `n`:
    a. For each column `j` from 1 to `i`:
        i. Print `*`
    b. Move to the next line
4. Stop

Program Code:

```c
#include <stdio.h>

int main() {
    int i, j, n;

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

    for (i = 1; i <= n; i++) {
        for (j = 1; j <= i; j++) {
            printf("* ");
        }
        printf("\n");
    }

    return 0;
}
```

Sample Output:
Enter number of rows: 4

*

* *

* * *

* * * *

Lab Exercise 2: Generate Fibonacci Series

Objective: To write a C program that generates and displays the Fibonacci series up to n terms.

Problem Statement: Write a C program to print the Fibonacci series for the first n terms, where the user enters n.

Algorithm:

1. Start
2. Read the number of terms n
3. Initialize first = 0, second = 1
4. Repeat until i < n:
5. a. Print first
6. b. next = first + second
7. c. first = second, second = next
8. Stop

Program Code:

```c
#include <stdio.h>

int main() {
    int n, first = 0, second = 1, next, i;
    printf("Enter the number of terms: ");
    scanf("%d", &n);
    printf("Fibonacci Series: ");
    for (i = 0; i < n; i++) {
        printf("%d ", first);
        next = first + second;
        first = second;
        second = next;
    }
    return 0;
}
```

Sample Output:

Enter the number of terms: 7
Fibonacci Series: 0 1 1 2 3 5 8

Lab Exercise 3: Find the Sum of Digits of a Number

Objective: To calculate the sum of digits of a given number.

Problem Statement:** Write a C program to find the sum of all digits of an integer entered by the user.

Algorithm:

1. Start
9. Read the number n
10. Initialize sum = 0
11. Repeat while n != 0:
12. a. digit = n % 10
13. b. sum = sum + digit
14. c. n = n / 10
15. Print sum
16. Stop

Program Code:

```c
#include <stdio.h>

int main() {
    int n, sum = 0, digit;
    printf("Enter a number: ");
    scanf("%d", &n);
    while (n != 0) {
        digit = n % 10;
        sum += digit;
        n /= 10;
    }
    printf("Sum of digits = %d\n", sum);
    return 0;
}
```

Sample Output:

Enter a number: 1234
Sum of digits = 10

Lab Exercise 4: Print All Prime Numbers in a Given Range

Objective: To identify and print all prime numbers within a given range.

Problem Statement: Write a C program to display all prime numbers between two integers entered by the user.

Algorithm:

1. Start
17. Read low and high
18. For each number i in [low, high]:
19. a. Skip if i <= 1
20. b. Check divisibility from 2 to √i
21. c. If not divisible by any, print i
22. Stop

Program Code:

```c
#include <stdio.h>

int main() {
    int low, high, i, j, flag;
    printf("Enter lower and upper limits: ");
    scanf("%d %d", &low, &high);
    printf("Prime numbers between %d and %d are: ", low, high);
    for (i = low; i <= high; i++) {
        if (i <= 1)
            continue;
        flag = 0;
        for (j = 2; j * j <= i; j++) {
            if (i % j == 0) {
                flag = 1;
                break;
            }
        }
        if (flag == 0)
            printf("%d ", i);
    }
    return 0;
}
```

Sample Output:

Enter lower and upper limits: 10 30
Prime numbers between 10 and 30 are: 11 13 17 19 23 29

Lab Exercise 5: Reverse a Given Number

Objective: To find and display the reverse of an integer.

Problem Statement: Write a C program to reverse the digits of an integer entered by the user.

Algorithm:

1. Start
23. Read n
24. Initialize reversed = 0
25. Repeat while n != 0:
26. a. digit = n % 10
27. b. reversed = reversed * 10 + digit
28. c. n = n / 10
29. Print reversed
30. Stop

Program Code:

```c
#include <stdio.h>

int main() {
    int n, reversed = 0, digit;
    printf("Enter a number: ");
    scanf("%d", &n);
    while (n != 0) {
        digit = n % 10;
        reversed = reversed * 10 + digit;
        n /= 10;
    }
    printf("Reversed number = %d\n", reversed);
    return 0;
}
```

Sample Output:

Enter a number: 12345
Reversed number = 54321

Lab Exercise 6: Check Whether a Number is Palindrome or Not

Objective: To determine whether a given number is a palindrome.

Problem Statement: Write a C program to check whether a number is palindrome or not.

Algorithm:

1. Start
31. Read the number n
32. Store original = n
33. Reverse the number
34. If original == reversed Palindrome
35. Else Not Palindrome
36. Stop

Program Code:

```c
#include <stdio.h>

int main() {
    int n, original, reversed = 0, digit;
    printf("Enter a number: ");
    scanf("%d", &n);
    original = n;
    while (n != 0) {
        digit = n % 10;
        reversed = reversed * 10 + digit;
        n /= 10;
    }
    if (original == reversed)
        printf("%d is a palindrome.\n", original);
    else
        printf("%d is not a palindrome.\n", original);
    return 0;
}
```

Sample Output:

Enter a number: 1221
1221 is a palindrome.