📘 C Programming Basics

🧪 Lab Session: Introduction to C Language (Conditional Statements, Loops & Functions)

Course: Operating Systems
Date: 19.05.2025

Introduction to Conditional Statements

Conditional statements allow the program to take different actions based on specific conditions. This includes decision-making structures like if-else, ternary operators, and switch-case structures.

1. if-else Statement

The if-else statement lets you execute certain code when a condition is true and different code when it's false. It is useful for branching logic based on conditions.

```
if (condition) {
   // body of if statement
} else {
   // body of else statement
}
```

Example:
Check if a person is eligible to vote (age >= 18)

```
#include<stdio.h>
int main() {
   int age;
   printf("Enter your age: ");
   scanf("%d", &age);

   if (age >= 18) {
      printf("You are eligible for voting\n");
   } else {
        printf("Not eligible for voting\n");
    }
    return 0;
}
```

Fedora Output:

```
[2021ict108@fedora ~]$ gcc eligible.c -o eligible
[2021ict108@fedora ~]$ ./eligible
Enter your age: 24
You are eligible for voting
[2021ict108@fedora ~]$ ./eligible
Enter your age: 13
Not eligible for voting
```

---

2. Ternary Operator

The ternary operator is a shorthand way of writing simple if-else conditions. It is ideal for conditions that assign or return values based on a boolean expression.

```
test_condition ? expression1 : expression2;
```

Example:

```
#include<stdio.h>
int main() {
    int age;
    printf("Enter your age: ");
    scanf("%d", &age);

    (age >= 18) ? printf("You are eligible\n") : printf("You are not eligible\n");
    return 0;
}
```

Fedora Output:

```
[2021ict108@fedora ~]$ gcc ternary.c -o ternary

[2021ict108@fedora ~]$ ./ternary

Enter your age: 25

You are eligible

[2021ict108@fedora ~]$ ./ternary

Enter your age: 14

You are not eligible
```

Introduction to switch Statement

Example 1: Day of the Week

The switch-case structure is used here to execute different outputs depending on the numeric input value. It is particularly helpful for discrete values like menu options or days of the week.

```
#include <stdio.h>

int main() {

  int day;

  printf("Enter the number between 1 to 7: ");

  scanf("%d", &day);


  switch (day) {

    case 1: printf("Today is Sunday!\n"); break;

    case 2: printf("Today is Monday!\n"); break;

    case 3: printf("Today is Tuesday!\n"); break;

    case 4: printf("Today is Wednesday!\n"); break;

    case 5: printf("Today is Thursday!\n"); break;
```

```
        case 6: printf("Today is Friday!\n"); break;
        case 7: printf("Today is Saturday!\n"); break;
        default: printf("Invalid input!\n");
    }
    return 0;
}
```

Fedora Output:

```
[2021ict108@fedora ~]$ gcc weekday.c -o weekday
[2021ict108@fedora ~]$ ./weekday
Enter the number between 1 to 7: 5
Today is Thursday!
```

Example 2: Life Path Number Astrology

```
#include <stdio.h>
int main() {
    int date, a, b, c;
    printf("Enter your birth date (1 to 31): ");
    scanf("%d", &date);

    if (date < 1 || date > 31) {
        printf("Invalid date input!\n");
        return 0;
    }

    a = date % 10;
    b = date / 10;
    c = a + b;
    printf("Date: %d\n", date);
    printf("Life Path Number: %d\n", c);

    switch (c) {
        case 1: printf("Lucky\n"); break;
        case 2: printf("Carefully do your work\n"); break;
        case 3: printf("Stronger\n"); break;
        case 4: printf("Happy\n"); break;
        case 5: printf("Can get help\n"); break;
        case 6: printf("Doubt\n"); break;
        case 7: printf("Sad\n"); break;
        case 8: printf("Like\n"); break;
        case 9: printf("Courage\n"); break;
        default: printf("Invalid life path number\n");
    }
    return 0;
}
```

Fedora Output:

```
[2021ict108@fedora ~]$ gcc astrology.c -o astrology
[2021ict108@fedora ~]$ ./astrology
Enter your birth date (1 to 31): 30
Date: 30
Life Path Number: 3
Stronger
```

Looping & Iteration Examples

Example 3: Sum and Multiplication using for loop

A for loop is used to perform repeated operations. Here, it calculates the sum and product of numbers from 1 to 5.

```
#include <stdio.h>

int main() {

    int i, sum = 0, mul = 1;

    for (i = 1; i <= 5; i++) {

        sum += i;

        mul *= i;

    }

    printf("Summation = %d\n", sum);

    printf("Multiplication = %d\n", mul);

    return 0;

}
```

Fedora Output:

```
Summation = 15

Multiplication = 120
```

Example 4: Print 1 to 10 using while loop

```
#include <stdio.h>

int main() {

    int i = 1;

    while (i <= 10) {

        printf("%d\n", i);

        i++;

    }
    return 0;
}
```

Fedora Output:

```
1
2
3
4
5
6
7
8
9
10
```

Example 5: Fibonacci Series

A Fibonacci series is a sequence where each term is the sum of the two previous terms. This example prints the first N numbers of the series using a loop.

```
#include <stdio.h>
int main() {
    int n, i;
    long long t1 = 0, t2 = 1, nextTerm;
    printf("Enter the number of terms: ");
    scanf("%d", &n);

    printf("Fibonacci Series: ");
    for (i = 1; i <= n; i++) {
        printf("%lld ", t1);
        nextTerm = t1 + t2;

        t1 = t2;

        t2 = nextTerm;

    }

    return 0;

}
```

Fedora Output:

```
Enter the number of terms: 5

Fibonacci Series: 0 1 1 2 3
```

Example 6: Factorial Calculation

```
#include <stdio.h>

long long factorial(int n) {

    long long fact = 1;

    for (int i = 1; i <= n; i++)

        fact *= i;

    return fact;

}

int main() {

    int num;

    printf("Enter a number: ");

    scanf("%d", &num);

    printf("Factorial: %lld\n", factorial(num));

    return 0;

}
```

Fedora Output:

```
Enter a number: 5
```

```
Factorial: 120
```

Example 7: String Concatenation

```
#include <stdio.h>
#include <string.h>
int main() {
    char str1[100], str2[100], result[200];
    printf("Enter first string: ");
    fgets(str1, sizeof(str1), stdin);
    printf("Enter second string: ");
    fgets(str2, sizeof(str2), stdin);

    str1[strcspn(str1, "\n")] = '\0';
    str2[strcspn(str2, "\n")] = '\0';

    strcpy(result, str1);
    strcat(result, str2);
    printf("Concatenated string: %s\n", result);
    return 0;
}
```

Fedora Output:

```
Enter first string: Hello
Enter second string: World
Concatenated string: HelloWorld
```

Example 8: Binary to Decimal Conversion

```
#include <stdio.h>
int main() {
    int binary, decimal = 0, base = 1, rem;
    printf("Enter a binary number: ");
    scanf("%d", &binary);

    while (binary > 0) {
        rem = binary % 10;
        decimal += rem * base;
        binary /= 10;
        base *= 2;
    }
    printf("Decimal equivalent: %d\n", decimal);
    return 0;
}
```

Fedora Output:

```
Enter a binary number: 1010
Decimal equivalent: 10
```

Example 9: Find Max and Min in an Array

```
#include <stdio.h>
int main() {
    int n, i;
    printf("Enter number of elements: ");
    scanf("%d", &n);
    int arr[n];

    printf("Enter %d elements: ", n);
    for (i = 0; i < n; i++)
```

```
        scanf("%d", &arr[i]);

    int max = arr[0], min = arr[0];
    for (i = 1; i < n; i++) {
        if (arr[i] > max) max = arr[i];
        if (arr[i] < min) min = arr[i];
    }
    printf("Max = %d, Min = %d\n", max, min);
    return 0;
}
```

Fedora Output:

```
Enter number of elements: 5
Enter 5 elements: 2 4 6 1 9
Max = 9, Min = 1
```

Example 10: Pascal's Triangle

Pascal's Triangle is a triangular arrangement of numbers. Each number is the sum of the two directly above it. This program generates it using nested loops.

```
#include <stdio.h>
int main() {
    int rows, coef = 1, space, i, j;
    printf("Enter number of rows: ");
    scanf("%d", &rows);
    for (i = 0; i < rows; i++) {
        for (space = 1; space <= rows - i; space++)
            printf("  ");
        for (j = 0; j <= i; j++) {
            if (j == 0 || i == 0)
                coef = 1;
            else
                coef = coef * (i - j + 1) / j;
            printf("%4d", coef);
        }
        printf("\n");
    }
    return 0;
}
```

Fedora Output:

```
Enter number of rows: 5
           1
         1   1
       1   2   1
     1   3   3   1
   1   4   6   4   1
```