**National Institute of Textile Engineering and Research (NITER)**

Nayarhat, Savar, Dhaka-1350

|  |  |
| --- | --- |
| **Experiment Number** | 01 |
| **Course Name** | Digital Logic Design |
| **Course Code** |  |
| **Lab Report Name** | **Understanding Arithmetic, Conditional, and Switch Statements in Programming** |
| **Date to Allotment:** |  |
| **Date of Submission** |  |

**Department of Computer Science and Engineering (CSE)**

|  |  |
| --- | --- |
| **Submitted By** | **Submitted To** |
| NAME :  ID :  Section : A1 |  |

**Understanding Arithmetic, Conditional, and Switch Statements in Programming**

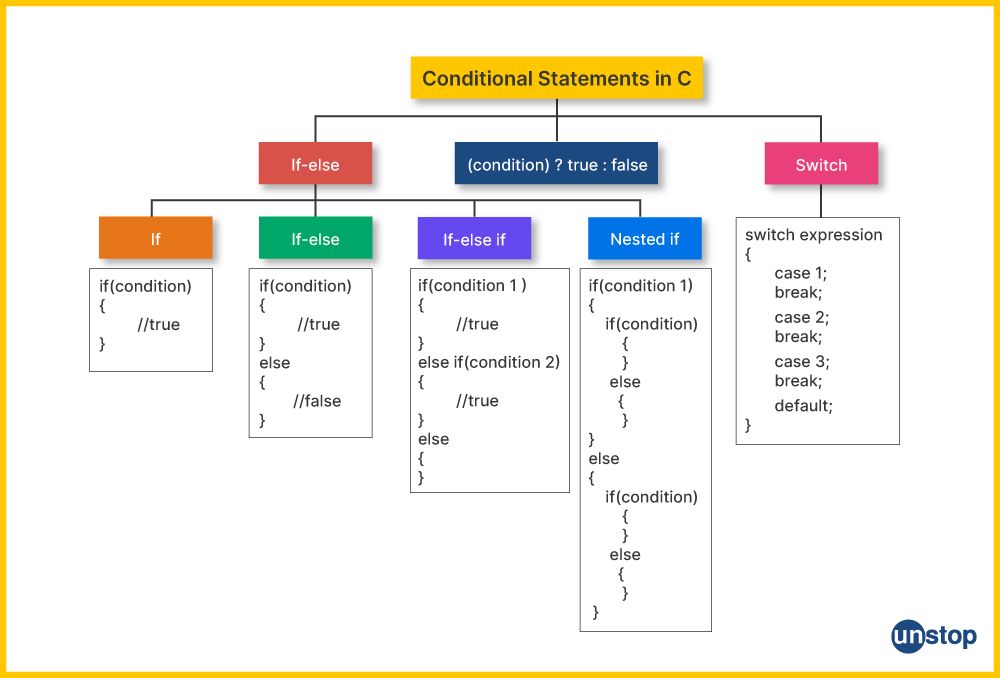
**1. Introduction**

In programming, arithmetic operations, conditional statements, and switch statements are foundational concepts that help in decision-making, computation, and control flow. These concepts are widely used across various programming languages like C++, Python, Java, and many others. This assignment will delve into each of these topics, discussing their importance and how they are implemented.

**2. What are Conditional Statements?**

A conditional statement, also known as an "if" statement, is a feature in programming that lets a program make choices based on the truth value of an expression. When a specified condition is met (true), the program executes a particular set of instructions. If the condition is not met (false), the program either skips that set of instructions or executes an alternative set.

**3. Types of Conditional Statements**

Most programming languages have several types of conditional statements. Here are the most commonly used ones:

**Arithmetic Operations**

Arithmetic operations are basic mathematical operations that involve numbers. In programming, arithmetic operations are used to manipulate data, perform calculations, and solve problems. The following are common arithmetic operations:

* Addition (+): Adds two numbers.
* Subtraction (-): Subtracts one number from another.
* Multiplication (\*): Multiplies two numbers.
* Division (/): Divides one number by another.
* Modulus (%): Returns the remainder of the division.

**Q: Write a C program to calculate the area of a triangle given its base and height**

#include<stdio.h>

// area of a tringle

int main() {

    float b, h;

    printf("Enter base and heigth :: ");

    scanf("%f %f", &b, &h);

    printf("Area of Traingle :: %f\n", 0.5 \* b \* h);

    return 0;

}

**Output:**

Enter base and height :: 5 10

Area of Triangle :: 25.000000

**Q: Write a C program to convert temperature from Fahrenheit to Celsius.**

#include<stdio.h>

int main() {

    printf("Farenhiet to Celcuis\n");

    printf("Enter Temp in Farenhiet::");

    float c, f;

    scanf("%f", &f);

    c = (f - 32.0) \* (5.0 / 9.0);

    printf("Celcuis temp :: %.2f\n", c);

    return 0;

}

**Output:**

Enter Temp in Fahrenheit:: 98.6

Celsius temp :: 37.00

Q: Write a C program to convert temperature from Celsius to Fahrenheit

#include<stdio.h>

int main() {

    printf("Celsius to Farenhiet\n");

    printf("Enter Temp in Celcuis::");

    float c, f;

    scanf("%f", &c);

    f = (9.0 / 5.0) \* c + 32.0;

    printf("Farenhiet temp :: %.2f\n", f);

    return 0;

}

**Output:**

Enter Temp in Celsius:: 37

Fahrenheit temp :: 98.60

**Q: Write a C program to calculate the square root of a given number.**

#include<stdio.h>

#include<math.h>

int main() {

    double n;

    printf("Enter n :: ");

    scanf("%lf", &n);

    printf("root(n) :: %.2lf\n", sqrt(n));

    return 0;

}

**Output:**

Enter n :: 16

root(n) :: 4.00

**Q: Write a C program to calculate the power of a number (an) given base a and exponent n.**

#include<stdio.h>

#include<math.h>

int main() {

    double a,n;

    printf("(a^n) :: Enter a and x :: ");

    scanf("%lf %lf", &a, &n);

    printf("Power :: %.2lf\n", pow(a, n));

    return 0;

}

**Output:**

Enter a and n :: 2 3

Power :: 8.00

**Q: Write a C program to evaluate the expression *Y = ax2+by+z*  given values for *a, x, b, y,* and *z*.**

#include<stdio.h>

int main() {

    double a, x, b, y, z;

    printf("Enter 'a, x, b, y, z' :: ");

    scanf("%lf %lf %lf %lf %lf", &a, &x, &b, &y, &z);

    double Y = a \* x \* x + b \* y + z;

    printf("Y = %lf\n", Y);

    return 0;

}

**Output:**

Enter 'a, x, b, y, z' :: 2 3 4 5 6

Y = 47.000000

**If Statements**

The **if statement** is the simplest form of conditional. It checks a condition and executes a block of code if the condition is true.

**Q: Write a C program to find the maximum value of 3 integers.**

#include<stdio.h>

int main() {

    int a, b, c;

    printf("Enter a,b,c ::");

    scanf("%d %d %d", &a, &b, &c);

    int max = a;

    if (b > max) {

        max = b;

    }

    if (c > max) {

        max = c;

    }

    printf("Max :: %d\n", max);

    return 0;

}

**Output:**

Enter a, b, c :: 10 20 15

Max :: 20

**If-Else Statements**

The **if-else** structure adds more flexibility by allowing multiple conditions. The program checks each condition in sequence until it finds one that’s true.

**Q: Write a C program to check whether a number is divisible by 5 or 9.**

#include<stdio.h>

int main() {

    int n;

    scanf("%d", &n);

    if (n % 5 == 0 || n % 9 == 0) {

        printf("%d is divisable by 9 or 5\n", n);

    }

    else {

        printf("%d is not divisable by 9 or 5\n", n);

    }

    return 0;

}

**Output:**

Enter n :: 45  
45 is divisible by 9 or 5

**Q: Write a C program to check whether a number is divisible by 5 or 9.**

#include<stdio.h>

int main() {

    int y;

    scanf("%d", &y);

    if ((y % 4 == 0 && y % 100 != 0 )|| y % 400 == 0) {

        printf("Leap year\n");

    }

    else {

        printf("Not leap year\n");

    }

    return 0;

}

**Output:**

Enter year :: 2024  
Leap year

**Q: Write a C program to check whether a character is a vowel or not.**

#include<stdio.h>

int main() {

    char ch;

    scanf("%c", &ch);

    if (ch == 'a' || ch == 'e' || ch == 'i' || ch == 'o' || ch == 'u') {

        printf("%c is vowel\n", ch);

    }

    else {

        printf("%c is not a vowel\n", ch);

    }

    return 0;

}

**Output:**

Enter character :: a

a is vowel

**Nested If-else Statements**

A **nested if statement** is an if statement inside another if statement. It’s useful for checking multiple conditions in a specific hierarchy.

**Q: Write a C program to grade a number based on a given scale (A+, A, A-, B, C, D, F).**

#include<stdio.h>

int main() {

    int n;

    printf("Enter number :: ");

    scanf("%d", &n);

    if (n <= 100 && n >= 0) {

        if (n >= 80) printf("A+\n");

        else if (n >= 70) printf("A\n");

        else if (n >= 60) printf("A-\n");

        else if (n >= 50) printf("B\n");

        else if (n >= 40) printf("C\n");

        else if (n >= 33) printf("D\n");

        else printf("F\n");

    }

    else {

        printf("Wrong input!\n");

    }

}

**Output:**

Enter number :: 85

A+

**Switch Statement**

The switch statement is used when there are multiple possible values for a single variable, and we need to perform different actions based on the value of that variable. It provides a cleaner and more readable alternative to a series of if-else if statements.

**Q: Write a C program to perform basic arithmetic operations (+, -, \*, /, %) based on user input.**

#include<stdio.h>

int main() {

    int a, b;

    char ch;

    printf("Enter operator (+,-,\*,/,%%) :: ");

    scanf("%c", &ch);

    printf("Enter (a and b) :: ");

    scanf("%d %d", &a, &b);

    switch (ch)

    {

    case '+':

        printf("a + b = %d\n", a + b);

        break;

    case '-':

        printf("a - b = %d\n", a - b);

        break;

    case '\*':

        printf("a \* b = %d\n", a \* b);

        break;

    case '/':

        printf("a / b = %d\n", a / b);

        break;

    case '%':

        printf("a %% b = %d\n", a % b);

        break;

    default:

        printf("Invalid imput!\n");

        break;

    }

    return 0;

}

**Output:**

Enter operator (+,-,\*,/,%) :: +

Enter (a and b) :: 10 5

a + b = 15

**Conclusion**

Arithmetic operations, conditional statements, and switch statements are essential tools in programming that help with making decisions, controlling program flow, and performing calculations. Understanding how to use them is crucial for writing effective and efficient code. Mastering these basic concepts will enable programmers to solve complex problems and develop versatile applications. By combining arithmetic with conditional and switch statements, programmers can build programs that are both dynamic and responsive to different inputs.