
Session 3: Control Statements in C

Control statements in C are used to change the flow of execution in a program. They allow decision-making based on conditions. Common control statements are: if, if-else, if-else-if ladder, and switch-case.

1. if Statement

The 'if' statement executes a block of code only if the condition is true.

```
#include <stdio.h>
int main() {
    int num = 5;
    if (num > 0) {
        printf("Number is positive\n");
    }
    return 0;
}
```

2. if-else Statement

Provides an alternative block when the condition is false.

```
#include <stdio.h>
int main() {
    int num = 4;
    if (num % 2 == 0) {
        printf("Even\n");
    } else {
        printf("Odd\n");
    }
    return 0;
}
```

3. if-else-if Ladder

Used when multiple conditions need to be checked sequentially.

```
#include <stdio.h>
int main() {
    int marks = 75;
    if (marks >= 90)
        printf("Grade A\n");
    else if (marks >= 75)
```

```
        printf("Grade B\n");
    else if (marks >= 50)
        printf("Grade C\n");
    else
        printf("Fail\n");
    return 0;
}
```

4. switch-case Statement

Switch is used when we need to choose among multiple options. Each case must end with 'break' to prevent fall-through.

```
#include <stdio.h>
int main() {
    int choice = 2;
    switch(choice) {
        case 1: printf("Option 1 Selected\n"); break;
        case 2: printf("Option 2 Selected\n"); break;
        case 3: printf("Option 3 Selected\n"); break;
        default: printf("Invalid Choice\n");
    }
    return 0;
}
```

Common Questions and Answers

Q1: Difference between if-else and switch?
A: if-else is used for ranges/complex conditions; switch is better for fixed discrete values.

Q2: Why is 'break' needed in switch?
A: To stop execution from falling into the next case.

Q3: Can we nest if or switch?
A: Yes, nesting is allowed but should be avoided if too complex.

Q4: What happens if no case matches in switch?
A: The 'default' block executes if provided.