I'll explain the three main types of loops in programming with examples.

1. for loop

Used when you know exactly how many times you want to repeat.

Syntax:

```c
for (initialization; condition; increment/decrement) {
    // code to be executed
}
```

Example:

```c
#include <stdio.h>

int main() {
    // Print numbers from 1 to 5
    for (int i = 1; i <= 5; i++) {
        printf("%d ", i);
    }
    // Output: 1 2 3 4 5
    return 0;
}
```

2. while loop

Repeats as long as a condition is true.

Syntax:

```c
while (condition) {
    // code to be executed
}
```

Example:

```c
```

```c
#include <stdio.h>

int main() {
    int count = 1;

    // Print numbers from 1 to 5
    while (count <= 5) {
        printf("%d ", count);
        count++;
    }
    // Output: 1 2 3 4 5
    return 0;
}
```

3. do-while loop

Executes at least once, then repeats as long as condition is true.

Syntax:

```c
do {
    // code to be executed
} while (condition);
```

Example:

```c
#include <stdio.h>

int main() {
    int number;

    // Ask for input at least once
    do {
        printf("Enter a positive number: ");
        scanf("%d", &number);
    } while (number <= 0);

    printf("Thank you! You entered: %d\n", number);
    return 0;
}
```

```

Key Differences:

Loop Type When to Use Minimum Executions
for Known number of iterations 0
while Unknown iterations, check condition first 0
do-while Unknown iterations, execute at least once 1

Practical Example Comparing All Three:

```c
#include <stdio.h>

int main() {
    int n = 5;

    printf("for loop: ");
    for (int i = 1; i <= n; i++) {
        printf("%d ", i);
    }

    printf("\nwhile loop: ");
    int j = 1;
    while (j <= n) {
        printf("%d ", j);
        j++;
    }

    printf("\ndo-while loop: ");
    int k = 1;
    do {
        printf("%d ", k);
        k++;
    } while (k <= n);

    return 0;
}
```

Output:

```
for loop: 1 2 3 4 5
```
```

while loop: 1 2 3 4 5
do-while loop: 1 2 3 4 5