Module: 2

Day2:
Trainer: Asif Nafees
while:
the while loop is used to repeatedly execute a block of code as long as a specified condition is true. It's a type of entry-controlled loop, meaning that the condition is evaluated before entering the loop body.

Syntax of the while Loop:
```
while (condition) {
   // Code to be executed
}
```

Key Points:
1. Condition: The while loop evaluates the condition at the start of each iteration. If the condition evaluates to true (non-zero), the loop body executes. If the condition evaluates to false (zero), the loop terminates.
2. Code Block: The code block inside the loop will continue to execute as long as the condition is true.
3. Indefinite Looping: If the condition never becomes false, the loop will run indefinitely (infinite loop).

Example of a while Loop in C:
```
#include <stdio.h>
int main() {
   int count = 1;  // Initialize a counter variable
   // while loop continues as long as count is less than or equal to 5
   while (count <= 5) {
      printf("Count is: %d\n", count);
      count++;  // Increment count to avoid infinite loop
   }
   return 0;
}
```
Output:
```
Count is: 1
Count is: 2
Count is: 3
Count is: 4
Count is: 5
```

Infinite Loop in C:
An infinite loop occurs if the condition in the while loop always remains true. For example:
```
#include <stdio.h>
int main() {
    while (1) {  // Infinite loop because condition is always true
        printf("This loop runs forever!\n");
    }
    return 0;
}
```

Using break and continue in while Loops:
break: The break statement can be used to exit the loop prematurely, even if the condition is still true.
continue: The continue statement skips the remaining code in the loop body for the current iteration and moves to the next iteration.
Example with break:

```
#include <stdio.h>
int main() {
    int i = 1;
    while (i <= 10) {
        printf("%d\n", i);
        if (i == 5) {
            break;  // Exits the loop when i equals 5
        }
        i++;
    }
    return 0;
}
```

Output:

```
1
2
3
4
5
```

Example with continue:

```
#include <stdio.h>
int main() {
    int i = 0;
    while (i < 5) {
        i++;
        if (i == 3) {
            continue;  // Skips printing when i equals 3
        }
        printf("%d\n", i);
    }
    return 0;
}
```

Output:

```
1
2
4
5
```

Infinite Loops: If the condition never becomes false, the loop will continue indefinitely. Always ensure that the loop has a condition that can eventually evaluate to false.

Example of a potential infinite loop:

```
int count = 1;
while (count <= 5) {
    printf("%d\n", count);
    // Forgot to increment 'count', causing an infinite loop
}
```