1. **Loop**: section of code that can be repeatedly executed

2. **do**
   
   body of loop
   
   while (condition); // do while the condition is true

   body of loop can be single statement or compound statement; Indent body

   **Post-test or exit-condition loop**: executes loop first then tests for loop exit
   Something must cause the condition to be false to exit the loop.

   ![Post-test or exit-condition loop diagram]

3. **while (condition) // while the condition is true**
   
   body of loop

   body of loop can be single statement or compound statement

   **Pre-test or entry-condition loop**: test condition before entering the loop

   ![Pre-test or entry-condition loop diagram]

4. **Terms:**
   
   **Control variable**: variable that controls whether the loop is executed or not
   
   **Loop Cycle**: the logical steps that may be repeated; is not necessarily the body of the loop
   
   **Input-bound Loop**: loop that exits at the end of the data
// Post-test loop: do...while example
#include <iostream> // for cin, cout
using namespace std;

int main()
{
    int num;
    char ans;
    // Pre-loop
    cout << "Welcome to number entry program!\n\n";
    do //Loop while user enters y
    {
        cout << "Enter a number: ";
        cin >> num;
        cout << "You entered " << num << "." << endl << endl;
        cout << "Another number? (Y/N): ";
        cin >> ans;
    }
    while (ans == 'y' || ans == 'Y'); // Control variable is ans
    cout << "\nEnd of Program\n"; // Post-loop
    return 0;
}

Sample Run:
Welcome to number entry program! (Pre-Loop)
Enter a number: 4 (Pass 1)
You entered 4.
Another number? (Y/N): y
Enter a number: 2 (Pass 2)
You entered 2.
Another number? (Y/N): y
Enter a number: 3 (Pass 3)
You entered 3.
Another number? (Y/N): n (Post-Loop)
End of Program

Terms:
Control variable: Variable that controls whether or not the loop is executed
Input-Bound Loop: Loop that exits at the end of the data
Start

"Welcome"

Input number

Output number

Another number ?

Yes

No

"End"

Stop
// Compute average grade for multiple students - Version 3
#include <iostream>    // for cin, cout
using namespace std;

int main()
{
    const double NUMBER_OF_TESTS = 3.0;    // Declared constant
    int grade1, grade2, grade3, grade_sum;
    double average_grade;
    char student_name[80];    // To store a string
    char ans;

    cout << "Midterm Grade Report\n";
    cout.setf(ios::fixed);
    cout.setf(ios::showpoint);
    cout.precision(1);

    do    // Do for each student
    {
        // Input student's data
        cout << "Enter student's name ==> ";
        cin.getline(student_name, 80);    // Reads thru \n, but doesn't store \n
        cout << "Enter first grade =====> ";
        cin >> grade1;
        cout << "Enter second grade ====> ";
        cin >> grade2;
        cout << "Enter third grade =====> ";
        cin >> grade3;

        // Compute sum and average of the grades
        grade_sum = grade1 + grade2 + grade3;
        average_grade = grade_sum / NUMBER_OF_TESTS;

        // Output the student report
        cout << "Student's name: " << student_name
             << "\nTest grades: " << grade1 << " " << grade2 << " " << grade3
             << "\nGrade sum: " << grade_sum
             << "\nGrade average: " << average_grade;

        if (average_grade >= 80.0)
            cout << " A or B Average\n";
        else
            cout << " C Average or lower\n";
        cout << "Another student? (Y or N): ";
        cin >> ans;
        cin.get();    // Discard newline before next getline()
    } while (ans == 'y' || ans == 'Y');

    cout << "End of Midterm Grade Report\n";
    return 0;    // 0 indicate successful program termination
Sample Run:
Midterm Grade Report

Enter student's name ==> Carol Wong
Enter first grade =====> 100
Enter second grade =====> 76
Enter third grade =====> 83

Student's name: Carol Wong
Test grades: 100 76 83
Grade sum: 259
Grade average: 86.3  A or B Average

Another student? (Y or N): Y

Enter student's name ==> Jim Garcia
Enter first grade =====> 100
Enter second grade =====> 100
Enter third grade =====> 100

Student's name: Jim Garcia
Test grades: 100 100 100
Grade sum: 300
Grade average: 100.0  A or B Average

Another student? (Y or N): y

Enter student's name ==> Bich Tran
Enter first grade =====> 0
Enter second grade =====> 100
Enter third grade =====> 50

Student's name: Bich Tran
Test grades: 0 100 50
Grade sum: 150
Grade average: 50.0  C Average or lower

Another student? (Y or N): n

End of Midterm Grade Report

Variables during the 3 iterations:

<table>
<thead>
<tr>
<th>ans</th>
<th>student_name</th>
<th>average_grade</th>
</tr>
</thead>
<tbody>
<tr>
<td>Y</td>
<td>Carol Wong</td>
<td>86.3333333333</td>
</tr>
<tr>
<td>y</td>
<td>Jim Garcia</td>
<td>100.0</td>
</tr>
<tr>
<td>n</td>
<td>Bich Tran</td>
<td>50.0</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>grade_sum</th>
<th>grade_grade3</th>
<th>grade_grade2</th>
<th>grade_grade1</th>
</tr>
</thead>
<tbody>
<tr>
<td>259</td>
<td>83</td>
<td>76</td>
<td>100</td>
</tr>
<tr>
<td>300</td>
<td>100</td>
<td>100</td>
<td>100</td>
</tr>
<tr>
<td>150</td>
<td>50</td>
<td>100</td>
<td>0</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>main() NUMBER_OF_TESTS</th>
<th>pass 1</th>
<th>pass 2</th>
<th>pass 3</th>
</tr>
</thead>
<tbody>
<tr>
<td>3.0</td>
<td>3.0</td>
<td>3.0</td>
<td>3.0</td>
</tr>
</tbody>
</table>
// Pre-test loop: while example

#include <iostream> // for cin, cout

using namespace std;

int main()
{
    int num;

    // Pre-loop
    cout << "Welcome to number entry program!\n\n";
    cout << "Enter first number (negative to Quit): ";

    cin >> num; // Start first cycle; Priming input

    // Control variable is num
    while (num >= 0)
    {
        cout << "You entered " << num << "." << endl;
        cout << "Enter next number: ";
        cin >> num; // Start next cycle
    }

    // Negative value is trailer value
    cout << "\nEnd of Program\n"; // Post-loop
    return 0;
}

Sample Run:
Welcome to number entry program! (Pre-Loop)

Enter first number (negative to Quit): 4
You entered 4. (Pass 1)
Enter next number: 2
You entered 2. (Pass 2)
Enter next number: 3
You entered 3. (Pass 3)
Enter next number: -1 (Post-Loop)
End of Program

Term:

Trailer data or Sentinel: Last data item which marks the end of the data
Start

"Welcome"

Input first number

number >= 0 ?

true

Output number

false

"End"

Stop

Input next number
// Using break to exit a loop
#include <iostream>    // for cin, cout
using namespace std;

int main()
{
    int num;               // Pre-loop
    cout << "Welcome to number entry program!\n\n";
    while (1) // Endless loop; 1 is true
    {
        cout << "Enter a number (negative to Quit): ";
        cin >> num;
        if (num < 0)
            break;    // Exit loop
        cout << "You entered " << num << "." << endl << endl;
    }
    cout << \nEnd of Program
;                    // Post-loop
    return 0;
}

Sample Run:
Welcome to number entry program!    (Pre-Loop)
Enter a number (negative to Quit): 4    (Pass 1)
You entered 4.
Enter a number (negative to Quit): 2    (Pass 2)
You entered 2.
Enter a number (negative to Quit): 3    (Pass 3)
You entered 3.
Enter a number (negative to Quit): -1    (Pass 4)
(End of Program (Post-Loop))
Start

"Welcome"

Input number

Negative ?

true

Output number

false

"End"

Stop
// Counting and Accumulating

#include <iostream> // for cin, cout
using namespace std;

int main()
{
    int num;
    int count = 0; // Counter variable
    int sum = 0; // Accumulator variable

    cout << "Welcome to number entry program!\n\n";

    while (true) // bool constant
    {
        cout << "Enter a number (negative to Quit): ";
        cin >> num;

        if (num < 0)
            break;

        count = count + 1; // Update counter
        sum = sum + num; // Update accumulator

        cout << "You entered " << num << "." << endl << endl;
    }

    cout << "End of Program\n"; return 0;
}

Sample Run:
Welcome to number entry program!

Enter a number (negative to Quit): 4
You entered 4.
Enter a number (negative to Quit): 2
You entered 2.
Enter a number (negative to Quit): 3
You entered 3.
Enter a number (negative to Quit): -1
Your 3 numbers add up to 9.
End of Program

Note:
1. Counters and accumulators are:
   a. Initialized to 0 before the loop
   b. Updated within the loop
2. Better alternatives to line 23:    count += 1;    or    count++;    
   line 24:    sum += num;
1. Added with ANSI/ISO standard

2. English mathematician, George Boole, who developed a mathematical representation of the laws of logic.

   Boolean Algebra

3. **bool** variable has one of the two values: **true** or **false**

4. **bool** sent, received = true;

   **bool** constants: **true** and **false**

5. **bool** true and false can be promoted to **int** with true converting to 1 and false to 0

   ```
   int yes = true;  // yes initialized to 1
   int no = false;  // no initialized to 0
   ```

6. Numeric values can be converted to **bool**

   ```
   bool yea = 100;  // yea initialized to true
   bool nay = 0;    // nay initialized to false
   ```

7. ```
   cout << yea;  // bool type is displayed as 0 or 1
   cout << boolalpha << yea;  // displayed as false or true
   // manipulator converts bool to
   // "false" or "true" strings
   ```

8. **bool** b;

   ```
   cin >> b;  // 0 sets b to false; non-zero sets b to true
   ```

9. **sizeof (bool)** is 1
// Finding the smallest and largest values
#include <iostream> // for cin, cout
using namespace std;

int main()
{
    int num;
    int small = 101; // Initialize to out of range on high side
    int large = -1; // Initialize to out of range on low side

    cout << "Welcome to number entry program!\n"
        << "Valid range for numbers is 0 to 100, inclusive.\n\n";

    while (true)
    {
        cout << "Enter a number (negative to Quit): ";
        cin >> num;
        if (num < 0)
            break; // If trailer data, exit loop

        if (num < small)
            small = num; // Update smallest value

        if (num > large)
            large = num; // Update largest value

        cout << "\t\t\t\tYou entered " << num << "\n\n";
    }

    cout << "\nSmallest value: " << small << endl
        << "Largest value: " << large << endl;
    cout << "End of Program\n";
    return 0;
}

Sample Run:
Welcome to number entry program!
Valid range for numbers is 0 to 100, inclusive.
Enter a number (negative to Quit): 80
You entered 80.
Enter a number (negative to Quit): 2
You entered 2.
Enter a number (negative to Quit): 15
You entered 15.
Enter a number (negative to Quit): 90
You entered 90.
Enter a number (negative to Quit): -1
Smallest value: 2
Largest value: 90
End of Program
// Finding the smallest and largest values - Version 2
#include <iostream>  // for cin, cout
using namespace std;

int main()
{
    int num, small, large;  // No initial values
    cout << "Welcome to number entry program!
" << "Valid range for numbers is 0 to 2,147,483,647, inclusive.\n\n" << "Enter a number (negative to Quit): ";  // Start first cycle
    cin >> num;  // Priming input
    small = large = num;  // Multiple assignment
    while (num >= 0)
    {
        if (num < small)
            small = num;  // Update smallest value
        if (num > large)
            large = num;  // Update largest value
        cout << "\t\t\t\t\tYou entered " << num << "\n\n";
        cout << "Enter next number (negative to Quit): ";  // Start next cycle
        cin >> num;  // Duplicate input
    }
    cout << "\nSmallest value: " << small << endl
        << "Largest value: " << large << endl;
    return 0;
}

Sample Run:
Welcome to number entry program!
Valid range for numbers is 0 to 2,147,483,647, inclusive.
Enter a number (negative to Quit): 80
    You entered 80.
Enter next number (negative to Quit): 2
    You entered 2.
Enter next number (negative to Quit): 15
    You entered 15.
Enter next number (negative to Quit): 90
    You entered 90.
Enter next number (negative to Quit): -1
Smallest value: 2
Largest value: 90
End of Program
1. **Increment Operator**: Unary operator that increments operand by 1

   ```
   int a = 2;
   a++; // Now a is 3
   ```

   is equivalent to

   ```
   a = a + 1; or a += 1;
   ```

   a. Can only operate on variables:
   ```
   3++; // Invalid - operand is constant
   (a + 1)++; // Invalid - operand is expression
   ```

   b. Can only increase by 1

2. **Decrement Operator**: Unary operator that decrements operand by 1

   ```
   int b = 2;
   b--; // Now b is 1
   ```

   is equivalent to

   ```
   b = b - 1; or b -= 1;
   ```

   a. Can only operate on variables:
   ```
   3--; // Invalid
   (b + 1)--; // Invalid
   ```

   b. Can only decrease by 1

3. ```
   int a = 1, b = 1;
   int app, ppb, amm, mmb;
   ```

   // Postfix mode: use variable, then increment variable
   ```
   app = a++; // app = a;
   ```

   // Prefix mode: increment variable, then use variable
   ```
   ppb = ++b; // ++b;
   ```

   amm = a--; // amm = a;

   mmb = --b; // --b;
```