5. Repetition

6. Iteration

7. Other Conditional and Iterative Statements

The while Statement (1)

```
#include <iostream>
int main() {
   int count = 1; // Initialize counter
   while (count <= 5) {
      std::cout << count << '\n'; // Display counter, then
      count++; // Increment counter
   }
}
```

```
while( condition )
       statement
```

The while Statement (2)

```
#include <iostream>
int main() {
   char input;
   int count = 0;
   bool done = false;
   while (!done) {
      std::cout << count << '\n';
      std::cout << "Please enter \"Y\" to continue or \"N\" to quit:";
      std::cin >> input;
      if(input != 'Y' && input != 'y' && input != 'N' && input != 'n')
         std::cout << "\"" << input << "\""
                   << " is not a valid choice" << '\n';
      else if (input == 'Y' || input == 'y')
         count++;
      else if (input == 'N' || input == 'n')
         done = true;
   }
}
```


The while Statement (3)

```
#include <iostream>
int main() {
   int input = 0,      // Ensure the loop is entered
      sum = 0;         // Initialize sum

   // Request input from the user
   std::cout << "Enter numbers to sum, negative number ends list:";
   while (input >= 0) {       // A negative number exits the loop
      std::cin >> input;      // Get the value
      if (input >= 0)
         sum += input;        // Only add it if it is nonnegative
   }

   std::cout << "Sum = " << sum << '\n'; // Display the sum
}
```

The while Statement (4)

```
#include <iostream>
int main() {
   int input, sum = 0;
   std::cout << "Enter numbers to sum, type 'q' to end the list:";
   while (std::cin >> input) // ^Z(Windows), ^D(Unix)
      sum += input;

   std::cout << "Sum = " << sum << '\n';
}
```


The while Statement (5)

```
#include <iostream>
#include <iomanip>
// Print the powers of 10 from 1 to 1,000,000,000
int main() {
   int power = 1;
   while (power <= 1000000000) {
      // Right justify each number in a field 10 wide
      std::cout << std::setw(10) << power << '\n';
      power *= 10;
   }
}
```

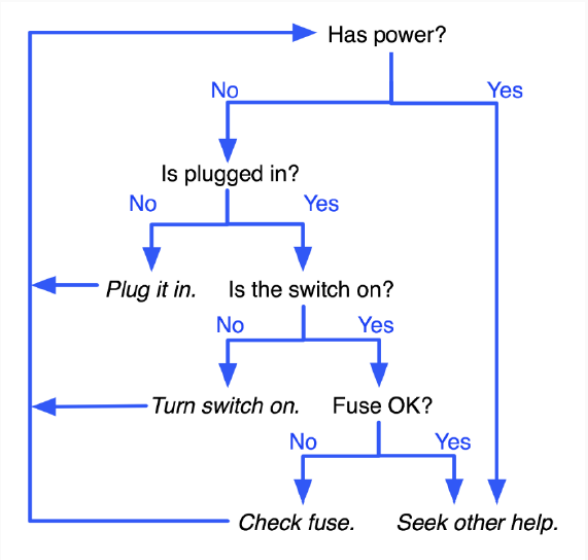
The while Statement (6)

```
#include <iostream>
#include <iomanip>
#include <locale>
// Print the powers of 10 from 1 to 1,000,000,000
int main() {
   int power = 1;
   std::cout.imbue(std::locale(""));
   while (power <= 1000000000) {
      // Right justify each number in a field 10 wide
      std::cout << std::setw(13) << power << '\n';
      power *= 10;
   }
}

// std::cout.imbue(std::locale("")); // global locale
// std::cout.imbue(std::locale("german")); // imbue locale
```


The while Statement (7)

```
// Decision tree for troubleshooting a computer system
```



Nested Loops

```
#include <iostream>
int main() {
   int size; // The number of rows and columns in the table
   std::cout << "Please enter the table size: ";
   std::cin >> size;
   // Print a size x size multiplication table
   int row = 1;
   while (row <= size) {      // Table has size rows.
      int column = 1;         // Reset column for each row.
      while (column <= size){ // Table has size columns.
         int product = row*column;    // Compute product
         std::cout << product << " "; // Display product
         column++; // Next element
      }
      std::cout << '\n'; // Move cursor to next row
      row++; // Next row
   }
}
```


Abnormal Loop Termination (1)

```
#include <iostream>
int main() {
   int input, sum = 0;
   std::cout << "Enter numbers to sum, negative number ends list:";
   while (true) {
      std::cin >> input;
      if (input < 0)
         break; // Exit loop immediately
      sum += input;
   }
   std::cout << "Sum = " << sum << '\n';
}
```

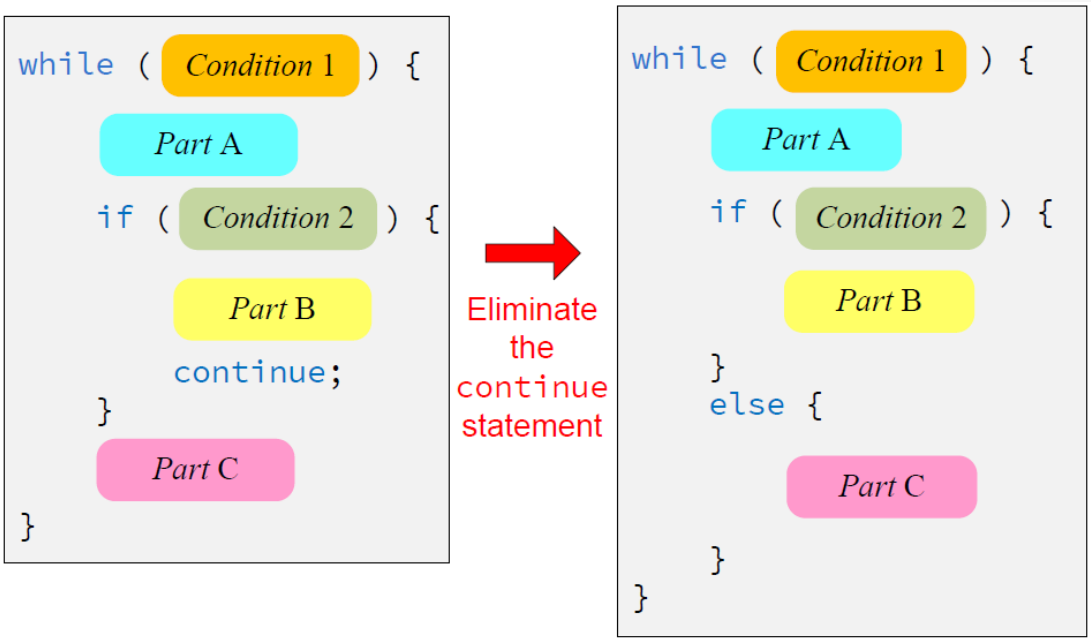
Abnormal Loop Termination (2)

```
#include <iostream>
int main() {
   int count = 1; // Initialize counter
top:
   if (count > 5)
      goto end;
   std::cout << count << '\n';        // Display counter, then
   count++;                           // Increment counter
   goto top;
end:
   ; // Target is an empty statement
}
```


Abnormal Loop Termination (3)

```
#include <iostream>
int main() {
   int input, sum = 0;        bool done = false;
   while (!done) {
      std::cout << "Enter positive integer (999 quits): ";
      std::cin >> input;
      if (input < 0) {
         std::cout << "Negative value " << input << " ignored\n";
         continue; // Skip rest of body for this iteration
      }
      if (input != 999) {
         std::cout << "Tallying " << input << '\n';
         sum += input;
      }
      else
         done = (input == 999); // 999 entry exits loop
   }
   std::cout << "sum = " << sum << '\n';
}
```

Abnormal Loop Termination (4)



Infinite Loops

```
while(true) {
   // Do something forever…
}

// Accidental infinite loop
#include <iostream>
int main() {
   int n = 1;
   const int MAX = 20;
   while (n <= MAX) {
      int factor = 1;
      std::cout << n << ": ";
      while (factor <= n)
         if (n % factor == 0) {
            std::cout << factor << " ";
            factor++;
         }
      std::cout << '\n';
      n++;
   }
}
```

```
1: 1
2: 1 2
3: 1
```

Iteration Examples (1)

```
#include <iostream>
int main() {
   int height;
   std::cout << "Enter height of tree: ";
   std::cin >> height;
   int row = 0;
   while (row < height) {
      int count = 0;
      while (count < height - row) {
         std::cout << " ";
         count++;
      }
      count = 0;
      while (count < 2*row + 1) {
         std::cout << "*";
         count++;
      }
      std::cout << '\n';
      row++;
   }
}
```

```
Enter height of tree: 5
     *
    ***
   *****
  *******
 *********
```


Iteration Examples (2)

```
#include <iostream>
int main() {
   int max_value;
   std::cout << "Display primes up to what value? ";
   std::cin >> max_value;
   int value = 2;
   while (value <= max_value) {
      bool is_prime = true;
      int trial_factor = 2;
      while (trial_factor < value) {
         if (value % trial_factor == 0) {
            is_prime = false;
            break;
         }
         trial_factor++;
      }
      if (is_prime)   std::cout << value << " ";
      value++;
   }
}
```

The switch Statement (1)

```
switch(integralExpression) {
   case integralConstant_1:
      statementSequence_1;
      break;
   case integralConstant_2:
      statementSequence_2;
      break;
   case integralConstant_n-2:
   case integralConstant_n-1:
      statementSequence_n-2_n-1;
      break;
   case integralConstant_n:
      statementSequence_n;
      break;
   [default:
      defaultStatementSequence;]
}
```


The switch Statement (2)

```
std::cin >> key;

switch (key) {
   case 'p':
   case 'P':
      std::cout << "You choose \"P\"\n";
      break;
   case 'q':
   case 'Q':
      done = true;
      break;
}
```

The Conditional Operator

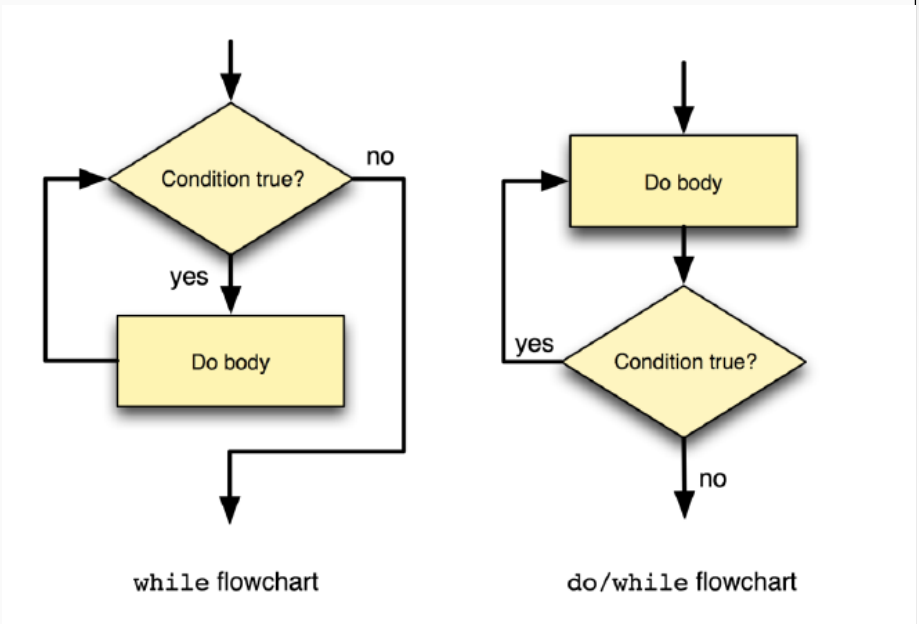
```
(condition)?expression_1:expression_2

x = (z != 0) ? y/z : 0;

(x < 0) ? ++n : n
```


The do/while Statement (1)

```
do
   statement;
while(condition);
```



The do/while Statement (2)

```
#include <iostream>
int main() {
   int in_value = -1;
   std::cout << "Please enter an integer in the range 0-10: ";
   while (in_value < 0 || in_value > 10)
      std::cin >> in_value;
   std::cout << "Legal value entered was " << in_value << '\n';
}
-----------------------------------------------------------
#include <iostream>
int main() {
   int in_value;
   std::cout << "Please enter an integer in the range 0-10: ";
   do
      std::cin >> in_value;
   while (in_value < 0 || in_value > 10);
   std::cout << "Legal value entered was " << in_value << '\n';
}
```


The for Statement (1)

```
for(initialization;condtion;modification)
   statement;



initialization
while(condtion) {
   statement;
   modification;
}


#include <iostream>
int main() {
   for (int count = 1; count <= 5; count++)
      std::cout << count << '\n';
}
```

The for Statement (2)

```
#include <iostream>
int main() {
   int max_value;
   std::cout << "Display primes up to what value? ";
   std::cin >> max_value;
   for (int value = 2; value <= max_value; value++) {
      bool is_prime = true;
      for (int trial_factor = 2;is_prime && trial_factor < value;
         trial_factor++)
         is_prime = (value % trial_factor != 0);

      if (is_prime)
         std::cout << value << " ";
   }
}
```


The for Statement (3)

```
for (int i = 0, j = 100; i < j; i++) {
   …
}

for (; i < 10; i++) {
   …
}

for (int i = 0; ; i++) {
   ...
}

for (int i = 0; i < 10; i++);

for ( ;; ) {
   …
}
```