4. Conditional Expressions

5. Conditional Execution

Type bool

- Boolean value
  - true, false
  - 1, 0
  - Non-zero integer, zero
- Relational operators
  - Result: Boolean value
  - ==, equal to
  - <, less than
  - >, greater than
  - <=, less than or equal to
  - >=, greater than or equal to
  - !=, not equal to

```
#include <iostream>
int main() {
   bool a = true;
   std::cout << a << '\n';
   a = false;
   std::cout << a << '\n';
   a = 0;
   std::cout << a << '\n';
   a = -10; // warning
   std::cout << a << '\n';
}

// std::cout << std::boolalpha;
```

Boolean Expressions

- Examples
  - `10 < 20`
  - `10 >= 20`
  - `x == 10`
  - `x != y`
  - `x+2 < y/10` `// (x+2) < (y/10)`
  - Precedence
    - ()
    - Unary
    - *, /, %
    - +, -
    - <, <=, >, >=
    - ==, !=
    - =

The Simple if Statement (1)

```
#include <iostream>
int main() {
   int dividend, divisor;
   // Get two integers from the user
   std::cout << "Please enter two integers to divide:";
   std::cin >> dividend >> divisor;
   // If possible, divide them and report the result
   if (divisor != 0)
      std::cout << dividend << "/" << divisor << " = "
        << dividend/divisor << '\n';
}

if(/*condition*/)
  // statement
```

The Simple if Statement (2)

```
if(x < 10);
   std::cout << "print" << std::endl;

//if(x < 10)
//   ; // null statement, empty statement
//std::cout << "print" << std::endl;

if(x = 10)
   std::cout << "print" << std::endl;

if(x =! 10)
   std::cout << "print" << std::endl;
```


Compound Statements (1)

```
#include <iostream>
int main() {
   int dividend, divisor, quotient;
   // Get two integers from the user
   std::cout << "Please enter two integers to divide:";
   std::cin >> dividend >> divisor;
   // If possible, divide them and report the result
   if (divisor != 0)
   {
      quotient = dividend / divisor;
      std::cout << dividend << "/" << divisor << " = "
        << quotient << '\n';
   }  // block, compound statement (zero or more statements)
}
```

Compound Statements (2)

```
if (x < 10)
   y = x;
   z = x + 5;

//if (x < 10)
//   y = x;
//z = x + 5;
```


The if/else Statement (1)

```
#include <iostream>
int main() {
   int dividend, divisor;
   std::cout << "Please enter two integers to divide:";
   std::cin >> dividend >> divisor;
   if (divisor != 0)
      std::cout << dividend << "/" << divisor << " = "
         << dividend/divisor << '\n';
   else
      std::cout << "Division by zero is not allowed\n";
}

if(/*condition*/)
  // statement1
else
  // statement2
```

The if/else Statement (2)

```
if (x == y)
   std::cout << x;
else {
   x = 0;
   std::cout << y;
}

if (x == y) {
   std::cout << x;
   x = 0;
}
else
   std::cout << y;
```

```
if (x == y) {
   std::cout << x;
   x = 0;
}
else {
   std::cout << y;
   y = 0;
}
```

```
if (x == 2)
   std::cout << x;
else
   ;

if (x == 2)
   std::cout << x;
else {
}

if (x == 2)
   std::cout << x;
```


The if/else Statement (3)

```
#include <iostream>
#include <iomanip>
int main() {
   double d1 = 1.11 - 1.10, d2 = 2.11 - 2.10;

   std::cout << "d1 = " << d1 << '\n';
   std::cout << "d2 = " << d2 << '\n';
   if (d1 == d2)
      std::cout << "Same\n";
   else
      std::cout << "Different\n";
   std::cout << "d1 = " << std::setprecision(20) << d1 << '\n';
   std::cout << "d2 = " << std::setprecision(20) << d2 << '\n';
}
//Different
//d1 = 0.010000000000000008882
//d2 = 0.009999999999997868372
```

Compound Boolean Expressions (1)

- Logical operators
  - Operand and Result: Boolean values
  - `&&, logical AND, binary`
  - `||, logical OR, binary`
  - `!, logical NOT, unary`
  - `Precedence`
    - `! > && > ||`

  - `x < y && x < z`
  - `// (x < y) && (x < z)`
  - `1 < x < 10`
  - `// (1 < x) < 10`
  - `// (1 < x) && (x < 10)`

Compound Boolean Expressions (2)

```
bool b;
int x = 10;
int y = 20;

b = (x == 10); // assigns true to b
b = (x != 10); // assigns false to b
b = (x == 10 && y == 20); // assigns true to b
b = (x != 10 && y == 20); // assigns false to b
b = (x == 10 && y != 20); // assigns false to b
b = (x != 10 && y != 20); // assigns false to b
b = (x == 10 || y == 20); // assigns true to b
b = (x != 10 || y == 20); // assigns true to b
b = (x == 10 || y != 20); // assigns true to b
b = (x != 10 || y != 20); // assigns false to b

//(x != y)
//!(x == y)
//(x < y || x > y)
```

Compound Boolean Expressions (3)

- Short-circuit evaluation
  - `A = true, B = true`
  - `A || B`
  - `A && B`

  - `// x = 0`
  - `(x != 0) && (z/x > 1)`

Compound Boolean Expressions (4)

- Precedence
  - `[unary] (post)++, (post)--, static_cast`
  - `[unary] (pre)++, (pre)--, !, +, -`
  - `[binary, left associativity] *, /, %`
  - `[binary, left associativity] +, -`
  - `[binary, left associativity] <<, >>`
  - `[binary, left associativity] >, <, >=, <=`
  - `[binary, left associativity] ==, !=`
  - `[binary, left associativity] &&`
  - `[binary, left associativity] ||`
  - `[binary, right associativity] =, +=, -=, *=, /=, %=`

Compound Boolean Expressions (5)

```
if (x == 1 || 2 || 3)
//if ((x == 1) || 2 || 3)
//if (x == 1 || x == 2 || x == 3)
```


Nested Conditionals (1)

```
#include <iostream>
int main() {
   int value;
   std::cout << "Please enter an integer value in the range 0...10: ";
   std::cin >> value;
   if (value >= 0) // First check
      if (value <= 10) // Second check
         std::cout << "In range";
   std::cout << "Done\n";
}
```

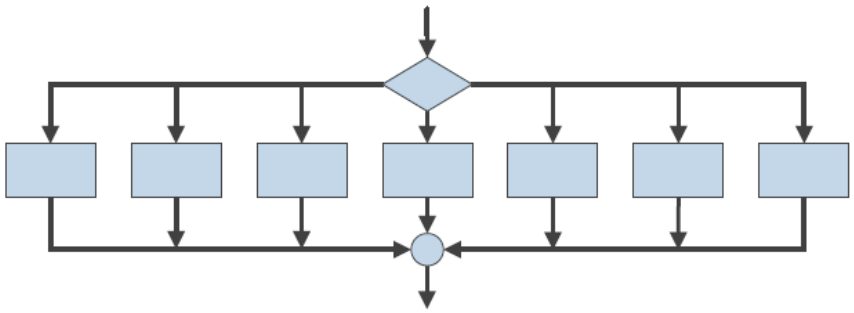
Nested Conditionals (2)

```
#include <iostream>
int main() {
   int value;
   std::cout << "Please enter an integer value in the range 0...10: ";
   std::cin >> value;
   if (value >= 0) // First check
      if (value <= 10) // Second check
         std::cout << value << " is acceptable\n";
      else
         std::cout << value << " is too large\n";
   else
      std::cout << value << " is too small\n";
}
```


Dangle else

```
#include <iostream>
int main() {
   int input;
   std::cin >> input;
   if (input >= 0)
      if(input < 2)
      std::cout << "zero, one\n";
   else
      std::cout << "negative\n";
}
```

Multi-way if/else Statements (1)



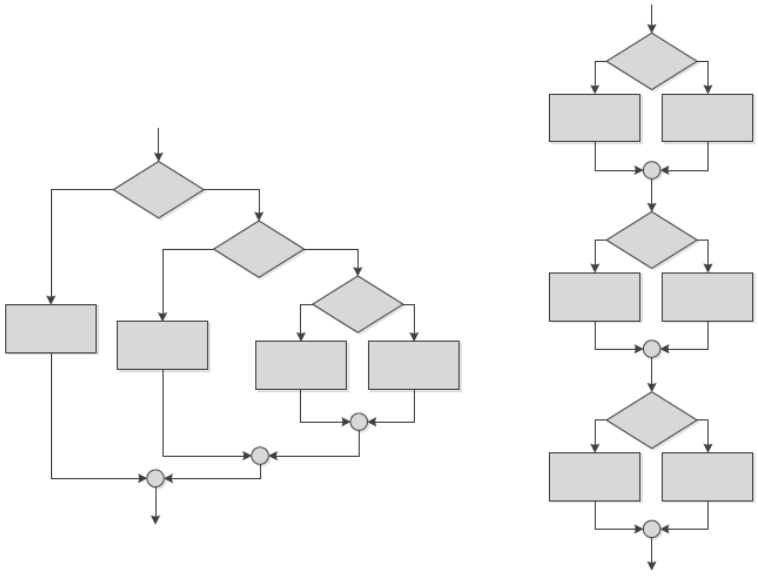
Multi-way if/else Statements (2)

```
#include <iostream>
int main() {
   int value;
   std::cout << "Please enter an integer in the range 0...5: ";
   std::cin >> value;
   if (value < 0)
      std::cout << "Too small";
   else
      if (value == 0)
         std::cout << "zero";
      else
         if (value == 1)
            std::cout << "one";
         else
            if (value == 2)
               std::cout << "two";
            else
               std::cout << "Too large";
   std::cout << '\n';
}
```

Multi-way if/else Statements (3)

```
#include <iostream>
int main() {
   int value;
   std::cout << "Please enter an integer in the range 0...5: ";
   std::cin >> value;
   if (value < 0)
      std::cout << "Too small";
   else if (value == 0)
      std::cout << "zero";
   else if (value == 1)
      std::cout << "one";
   else if (value == 2)
      std::cout << "two";
   else
      std::cout << "Too large";
   std::cout << '\n';
}
```


Multi-way if/else Statements (4)



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
   → expression_1(true) or expression_2(false)

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

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