ET-575 – Expressions - Handout

Number Format

- cout.setf(ios::fixed);
- cout.precision(x);
  - x specifies the number of decimal places
- **Ex 1a: number format**
- **Ex 1b: dollar convertor**

Operators

- **Operators and Operands**
  - x = y;
    - '=' is an operator
    - 'x' and 'y' are operands
    - '=' is a binary operator because it requires two operands.
  - -x;
    - '-' is an operator
    - 'x' is an operand
    - '-' is a unary operator because it requires one operand. (this is a negation operation)
  - x-y;
    - '-' is n operator
    - 'x' and 'y' are operands
      '-' is a binary operator because it requires two operands. (this is a subtraction operation)
  - **Ex 2: operators and operands**

- **Stream operators**
  - <<
    - insertion operator used with cout
    - bitwise left shift operator
    - can be used to multiply by multiples of 2
  - >>
    - extraction operator used with cin
    - bitwise right shift operator
    - can be used to divide by multiples of 2
  - **Ex 3: bitwise shift**

- **Basic Arithmetic Operators**
  - + addition
    - x = x + 5;
    - returns the sum of x and 5
  - – subtraction
    - x = x – 5;
    - returns the difference of x and 5
  - * multiplication
    - x = x * 5;
    - returns the multiple of x and 5
  - / division
    - all division by 0 equals undefined
    - x = x / 5;
    - returns the quotient of x and 5
    - regular division
      - 5 / 2 equals 2.5 or 2 remainder 1
      - 10 / 8 equals 1.25 or 1 remainder 2
    - integer division
      - 5 / 2 equals 2
      - 10 / 8 equals 1
  - % modulo division
    - integer operands only
    - x = x % 5;
    - returns the remainder of an integer division
      - 5 % 2 equals 1
      - 10 % 8 equals 2
  - Example usage of modulo division
    - to set a range of outputs

      1 % 5 equals 1
      2 % 5 equals 2
      3 % 5 equals 3
      4 % 5 equals 4
      5 % 5 equals 0
      6 % 5 equals 1
      7 % 5 equals 2
      8 % 5 equals 3
      9 % 5 equals 4
      10 % 5 equals 0
    - to determine if a number is even:
      - any even number % 2 must always equal 0
      - 10 % 2 equals 0
  - **Ex 4: types of division**
  - **Ex 8a: modulo division 1**
  - **Ex 8b: modulo division 2**
  - **Ex 8c: modulo division 3**
  - **Ex 8d: modulo division 4**

- **Shortcut Arithmetic Operators**
  - +=
    - x += 5;
    - equivalent to x = x + 5;
    - returns the sum of x and 5
  - -=
    - x -= 5;
    - equivalent to x = x – 5;
    - returns the difference of x and 5
  - *=
    - x *= 5;
    - equivalent to x = x * 5;
    - returns the multiple of x and 5
  - /=
    - x /= 5;
    - equivalent to x = x / 5;
    - returns the quotient of x and 5
  - %=
    - X %= 5;
    - equivalent to x = x % 5;
    - returns the remainder of x divided by 5
  - **Ex 5: shortcut operators**

- **Increment and Decrement Operators**
  - Prefix ++
    - prefix increment operator
    - y = ++x;
    - increment the value of x, return the value of x
  - Prefix --
    - Prefix decrement operator
    - Y = --x;
    - decrement the value of x, return the value of x
  - Postfix ++
    - Postfix increment operator
    - Y = x++;
    - create a copy of x, increment the value of x, and then return the value of the copy
  - Postfix --
    - Postfix decrement operator
    - Y = x--;
    - create a copy of x, decrement the value of x, and then return the value of the copy
  - **Ex 6: prefix vs postfix**

- **Relational Operators**
  - ==
    - Equals operator
    - (a == b)
    - returns a Boolean value
  - !=
    - Not-equals operator
    - (a != b)
    - returns a Boolean value
  - >
    - Greater-than operator
    - (a > b)
    - returns a Boolean value
  - <
    - Less-than operator
    - (a < b)
    - returns a Boolean value
  - >=
    - Greater-than or equal to operator
    - (a >= b)
    - returns a Boolean value
  - <=
    - Less-than or equal to operator
    - (a <= b)
    - returns a Boolean value
  - **Ex 7: assignment vs. equivalence**

Evaluation

- assignment
  - x = y = z is evaluated as x = (y = z)
- unary operators
  - x = y + -5 is evaluated as x = y + (-5)

- **Orders of Precedence**
  - Priority 1
    - . dot operator
    - [] array index
    - ( ) function call
    - n++ postfix increment operator
    - n-- postfix decrement operator
    - static_cast cast operation
  - Priority 2 (Right-to-Left)
    - ++n prefix increment operator
    - --n prefix decrement operator
    - ! not
    - - unary minus
    - + unary plus
  - Priority 3
    - * multiply
    - / divide
    - % modulo
  - Priority 4
    - + addition
    - - subtraction
  - Priority 5
    - << insertion operator
    - >> extraction operator
  - Priority 6
    - < less than
    - > greater than
  - Priority 7
    - == equivalence
    - != equivalence negation
  - Priority 8
    - && and
  - Priority 9
    - || or
  - Priority 10 (Right-to-Left)
    - = assignment
    - += add and assign
    - -= subtract and assign
    - *= multiply and assign
    - /= divide and assign
    - %/ modulo and assign
  - Priority 11
    - , comma

Random Numbers

- Pseudorandom numbers:
  - rand() – returns an int
  - Repetitively calling rand function will issue a sequence of random numbers based upon the original seed.
  - The mod operator can be used to limit the range of random numbers from 0 to n-1, such as rand( ) % n.
  - srand() – set the value of the starting seed
  - By updating or randomizing the seed in some way, it is possible to generate different pseudorandom sequences.

```
rand( )            // return a random number from 0 to RAND_MAX
rand( ) % 51       // will output a range of numbers from 0 to 50
srand(15)          // set the starting seed of random numbers
srand(time(NULL)) // set the starting seed using the current time
                      requires #include <time.h>
```

  - **Ex 9a: pseudorandom numbers**
  - **Ex 9b: seed generation**