- Today's topics:
  - ○ Type Checking
  - ○ Strings
  - ○ Input and Output
  - ○ Precedence and Boolean algebra
- Todo: Challenge1-Debug problem (Mon 8pm); T.C. and prelecture readings.

  Surprise Lecture Quiz (this/next week) - do the readings!

2. Write the following programs: (don't waste any time re-writing the entire text).

```
Enter a string with exactly 5 characters.
You typed:1234
Try again!
Enter a string with exactly 5 characters.
You typed:12345
Yes!
```

```
Please enter a string where the first and last letters are the same.
You typed "abbA"
You win!
```

```
Enter a word that includes the substring 'ting'
You entered: 'tingle'
Found 'ting' at position 1
```

1. Type Checking.

Are the following valid Java statements? If so, what will each print?

i) TextIO.putln("Result: " + 2 + 3);

ii) TextIO.putln(2 + 3 + "Result");

iii) int value = ((2 + (3 / 10) + 5.0) < 10) == true;

Useful String methods 'subroutines' from pre-lecture reading ch2.3)

s1. ______ (s2) returns true if s1 and s2 have of the same character sequence.

s1. ______ () the number of characters in s1.

s1. ______ (N) returns a *char* at position N

s1. ____________ (N,M) returns a string from N^{th} (inclusive) position up to but not including the M^{th} position.

s1.____________ (s2) returns an integer. If s2 occurs as a substring of s1, then the returned value is the starting position of that substring. Otherwise, the returned value is -1.

s1. _____________________ () returns a new string with lower case letters converted to upper case.

Useful TextIO methods (see ch2.4)

int guess= TextIO.getlnInt();   // Reads a value of type int.

double happiness = TextIO.getlnDouble();

String oneline = TextIO.getln(); // Reads an entire input line

TextIO.readFile("myfile.txt") // start reading from a file

TextIO.eof() // Returns true if there's no more to read

Java Operators *Highest*	Precedence
postfix (***Highest***)	expr++ expr--
unary	++expr --expr +expr -expr ~ **!**
multiplicative	* / %
additive	+ -
shift	<< >> >>>
relational	< > <= >=
equality	== !=
bitwise AND	&
bitwise exclusive OR	^
bitwise inclusive OR	|
logical **AND**	&&
logical **OR**	||
ternary	? :
assignment ***(Lowest)***	= += -= *= /= %= &= ^= |= <<= >>= >>>=

Java operator precedence

5 * 4 + 3 + 2 Evaluates to 20 + 3 + 2
(multiplication before addition)
20 + 3 + 2 Evaluates to... 23+2 25
(most operators work left to right)

3. Evaluate the following statement using Java's precedence rules

```
boolean r = ! true || false != false;
```

4. Evaluate

```
boolean r = 5 + 1 % 3 < 2 && 3 < 2 == false;
```

5. Truth tables - a peak at CS173 and introducing boolean algebra

```
// Input:
 boolean a = true;
 boolean b = true;
 boolean c = true;

//Output:
 boolean X= (a && b) || (a && c) || (b && c);
 boolean Y = a != b != c;
```

a	b	c	X	Y
0	0	0	0	0
0	0	1		
0	1	0		
0	1	1		
1	0	0		
1	0	1		
1	1	0		
1	1	1		

> Complete the truth table above-right for every possible combination of (a,b,c).

>Which of the following are valid descriptions of X,Y?

"True only when three input bits are set"

"True only when an odd number of input bits are set"

"True only when at least two input bits are set"

"True only when a equals c"

> What have X and Y expressions got to do with adding up two numbers (in binary)?