Exercise 1

Objects and expressions

This lesson is about reading and understanding Smalltalk expressions, and differentiating between different types of messages and receivers. Note that in the expressions you will be asked to read and evaluate, you can assume that the implementation of methods generally corresponds to what their message names imply (i.e. 2 + 2 = 4). **Exercise:** For each of the Smalltalk expressions below, fill in the answers:

```
3 + 4
```

- What is the receiver object?
- What is the message selector?
- What is/are the argument (s)?
- What is the message?
- What is the result returned by evaluating this expression?

```
Date today
```

- What is the receiver object?
- What is the message selector?
- What is/are the argument (s)?
- What is the message?
- What is the result returned by evaluating this expression?

```
anArray at: 1 put: 'hello'
```

- What is the receiver object?
- What is the message selector?
- What is/are the argument (s)?
- What is the message?
- What is the result returned by evaluating this expression?

Exercise: What kind of object does the literal expression 'Hello, Dave' describe? **Exercise:** What kind of object does the literal expression #Node1 describe? **Exercise:** What kind of object does the literal expression #(1 2 3) describe? **Exercise:** What can one assume about a variable named Transcript? **Exercise:** What can one assume about a variable named rectangle? **Exercise:** Examine the following expression :

```
| anArray |
anArray := #('first' 'second' 'third' 'fourth').
^anArray at: 2
```

What is the resulting value when it is evaluated (^means return)? What happens if you remove the ^? Explain

Exercise: Which sets of parentheses are redundant with regard to evaluation of the following

```
((3 + 4) + (2 * 2) + (2 * 3))

(x isZero)
   ifTrue: [....]
(x includes: y)
   ifTrue: [....]
```

Exercise: Guess what are the results of the following expressions

```
6 + 4 / 2
1 + 3 negated
1 + (3 negated)
2 raisedTo: 3 + 2
2 negated raisedTo: 3 + 2
```

Exercise: Examine the following expression:

25@50

- What is the receiver object?
- What is the message selector?
- What is/are the argument (s)?
- What is the message?
- What is the result returned by evaluating this expression?

Exercise: Examine the following expression and write down the sequence of steps that the Smalltalk system would take to execute the following expression: Date today daysInMonth **Exercise:** Examine the following expression and write down the sequence of steps that the Smalltalk system would take to execute the following expression: Transcript show: (45 + 9) printString **Exercise:** Examine the following expression and write down the sequence of steps that the Smalltalk system would take to execute the following expression: 5@5 extent: 6.0 truncated @ 7 **Exercise:** In lesson 1 we saw how to write strings to the Transcript, and how the message printString could be sent to any non-string object to obtain a string representation. Now write a Smalltalk expression to print the result of 34 + 89 on the Transcript. Test your code !

Exercise: Examine the block expression:

```
| anArray sum |
sum := 0.
anArray := #(21 23 53 66 87).
anArray do: [:item | sum := sum + item].
sum
```

What is the final result of sum ? How could this piece of code be rewritten to use explicit array indexing (with the method at:) to access the array elements ? Test your version. Rewrite this code using inject:into: