

**SE 2AA4, CS 2ME3 (Introduction to Software  
Development)**  
**Winter 2018**

## **25 English To Math**

Dr. Spencer Smith

Faculty of Engineering, McMaster University

March 14, 2018



## 25 English To Math

- Administrative details
- English and mathematics as languages
- Different world views
- Translating between languages
- Strategies
- Translator's glossary
- Exercise

# Administrative Details

- Midterm
  - ▶ Grades on Avenue
  - ▶ Average around 70%
  - ▶ Will change to out of 29 for final grade calculation
- A3
  - ▶ Part 1 - Specification: due 11:59 pm Mar 12
  - ▶ Part 1 - Solution: Mar 18
  - ▶ Part 2 - Code: due 11:59 pm Mar 26
  - ▶ Do not delete wss comments
  - ▶ Look at minor diffs in `se2aa4_cs2me3` repo for `spec.tex`
- A4
  - ▶ Your own design and specification
  - ▶ Due April 9 at 11:59 pm

# Natural Languages

- How many natural languages do you know?
- What natural languages do you know?

# English and Mathematics as Languages

See [Baber2002](#)

- English is a language
- So is Mathematics
- Both have
  - ▶ Rules of grammar (syntax)
  - ▶ Semantics
- When writing in any language, pay attention to grammar and semantics. Get both right.

# English and Mathematics: A Difference

- In English and other natural languages
  - ▶ Ambiguity desired, intentionally possible
  - ▶ Unambiguous statements almost impossible
- In Mathematics
  - ▶ Ambiguity not desired, intentionally prevented
  - ▶ Ambiguous statements almost impossible (even in probability theory, fuzzy logic)

# Mathematics and Engineering

- Therefore, mathematics is the language of engineering and science

# Correct Syntax for Mathematics

- Make sure the syntax of your mathematical expressions is correct
- Correct syntax does not guarantee correct semantics
- Incorrect syntax makes the mathematics ambiguous
- Example problems to watch for
  - ▶ Mismatch of types
    - ▶  $(p.xcoord() + width) \wedge p.xcoord()$
    - ▶  $\neg(\text{integer})$
    - ▶ Set of sequences accessed by  $[i]$
    - ▶  $\forall(i : \mathbb{N} \wedge j : \mathbb{N} \dots)$
  - ▶ Use of programming language notation in mathematics
    - ▶ Integer values instead of boolean
    - ▶ `length == MAX_SIZE`
  - ▶  $(x > 7) = \text{true}$  instead of  $(x > 7)$  (bad form)



# Different World Views Question

- Do the following languages have a static or dynamic world view?
  - ▶ Imperative programming language?
  - ▶ Mathematics?
  - ▶ Object oriented programming language?
  - ▶ Functional programming language?

# Different World Views

- English and other natural languages
  - ▶ Express both static and dynamic views
  - ▶ States and actions (verbs of being and action)
- Imperative programming languages
  - ▶ Primarily dynamic world view (changes)
- Functional programming languages
  - ▶ Static world view
- Mathematics
  - ▶ Static world view only
- Fundamental conceptual differences

# Static Versus Dynamic Views

- These very different world views pose a conceptual hurdle for the translator
- The translator must bridge the gap between
  - ▶ Dynamic and static view of problem statement
  - ▶ Dynamic world view of imperative and OO programming and
  - ▶ Purely static world view of mathematics
- Not hard, but requires conscious attention

# Translating Between Languages

- Lost in Translation
- Translating a statement from one language to another is a multistep (not single) process
  1. Statement in source language to a mental understanding of the meaning of the statement
  2. Reformulate mental understanding into target language view, concepts, culture
  3. Mental understanding of the meaning of the statement to a statement in the target language
- The first and last statement must mean the same

# Translators

- Knowing two languages: not enough to translate
- A good translator knows well
  - ▶ The two languages
  - ▶ AND the subject being translated
  - ▶ AND how to translate
- These three things are different

# Organization and Style

- When writing in English or any other natural language, one pays careful attention to
  - ▶ Organization of the essay, report, etc.
  - ▶ Style of expression
- When writing in Mathematics, to do the same:
  - ▶ Clear, complete, concise
  - ▶ KISS (Keep it Simple Sharp and Straightforward)
  - ▶ Understandable
  - ▶ Interesting

# Strategies

- Understand the meaning of the original
- Obtain all needed information
- Close the gap between the English text and mathematics
- Divide and conquer (complexity)

# Strategy: Understand the Original

- Describe specific instance of general problem
- Distinguish essentials from background
- Draw a diagram
- Express in intermediate or mixed language
- Identify object referred to
- Identify implicit (but false) “information”
- Identify missing information
- Identify relationships between essential objects
- Identify special cases



# Strategy: Obtain all Needed Information

- Ask the author of the task description
- Identify gaps in the description of the task
- Identify implicit “information”
- Ask if implicit “information” may be assumed
- Identify data present and ask about related details
- Ask if missing information is really needed
- Read **carefully, thoroughly, precisely**

# Strategy: Close Gap English – Mathematics

- Express implicit information explicitly
- Reduce vagueness and ambiguity
- Reword English text to be closer to mathematics (express in an intermediate, mixed language)

# Strategy: Divide and Conquer

- Construct a table
- Distinguish between specific cases
- Introduce an auxiliary mathematics function
- Modularize
- Example
  - ▶ Find the location in  $D$ (: sequence of  $T$ ) of a particular value  $key$ (:  $T$ )
  - ▶ What are the two cases?

# Strategy: Divide and Conquer

- Construct a table
- Distinguish between specific cases
- Introduce an auxiliary mathematics function
- Modularize
- Example
  - ▶ Find the location in  $D$ (: sequence of  $T$ ) of a particular value  $key$ (:  $T$ )
  - ▶ What are the two cases?
  - ▶  $key$  in  $D$ ,  $key$  not in  $D$

# Strategy: Divide and Conquer

- Construct a table
- Distinguish between specific cases
- Introduce an auxiliary mathematics function
- Modularize
- Example
  - ▶ Find the location in  $D$ (: sequence of  $T$ ) of a particular value  $key$ (:  $T$ )
  - ▶ What are the two cases?
  - ▶  $key$  in  $D$ ,  $key$  not in  $D$
  - ▶ How do you mathematically specify  $key$  is present in  $D$ ?

# Strategy: Divide and Conquer

- Construct a table
- Distinguish between specific cases
- Introduce an auxiliary mathematics function
- Modularize
- Example
  - ▶ Find the location in  $D$  (sequence of  $T$ ) of a particular value  $key$  ( $T$ )
  - ▶ What are the two cases?
  - ▶  $key$  in  $D$ ,  $key$  not in  $D$
  - ▶ How do you mathematically specify  $key$  is present in  $D$ ?
  - ▶  $\forall (i : \mathbb{N} | 1 \leq i \leq |D| : D[i] = key)$  (Baber starts his list index at 1)

# Strategy: Draw Diagrams, Describe Specific Instances of the Given Problem

- Graphical representations help understand the meaning of the message
- For specific instances, think of extreme cases first to simplify
  - ▶  $n = 0$
  - ▶  $n = 1$
  - ▶  $n = \text{inf}$
- Think of a normal sized problem, usually something like  $n \geq 3$
- You might want to write down truth tables (or Parnas tables)

# A Small Translator's Glossary

English	Mathematics
and, but	$\wedge$
or	$\vee$
for all, each, every, any	$\forall$ , $\wedge$ series, universal quantification
for no, none	$\forall$ , $\wedge$ series, universal quantification, with a negated assertion
there is (are), there exist(s), for some, at least one	$\exists$ , $\vee$ series, existential quantification



# A Small Translator's Glossary Continued

English	Mathematics
integer	$\dots \in \mathbb{Z}$
sorted	$\wedge_{i=0}^{n-2} A[i] \leq A[i+1],$ $\forall (i : \mathbb{N}   0 \leq i < n - 1 : A[i] \leq A[i+1])$
if (when, whenever) ... then ...	$\dots \rightarrow \dots$ , sometimes $\wedge$
search, find, equal, present	$=$
exchange, rearrange, different order, different sequence, merge, copy, sort	permutation

# Your Translator's Glossary

- A professional translator compiles his/her own translation glossary
  - ▶ Over time
  - ▶ Based on own accumulated experience
- You should too!

# Students with the same birthday

Consider a class with many students. The instructor bets the students that two or more students have the same birthday. Write a mathematical expression that is True if this condition is met, and False otherwise

- Identify all objects referred to in this problem statement

# Students with the same birthday

Consider a class with many students. The instructor bets the students that two or more students have the same birthday. Write a mathematical expression that is True if this condition is met, and False otherwise

- Identify all objects referred to in this problem statement
- Class? Students? Many? Instructor? Birthday?
- Distinguish between essential objects and background information

# Students with the same birthday

Consider a class with many students. The instructor bets the students that two or more students have the same birthday. Write a mathematical expression that is True if this condition is met, and False otherwise

- Identify all objects referred to in this problem statement
- Class? Students? Many? Instructor? Birthday?
- Distinguish between essential objects and background information
- How would we model the relationship between a student and their birthday? A set? A type? A variable? A relation? A function?

# Students with the same birthday

Consider a class with many students. The instructor bets the students that two or more students have the same birthday. Write a mathematical expression that is True if this condition is met, and False otherwise

- Identify all objects referred to in this problem statement
- Class? Students? Many? Instructor? Birthday?
- Distinguish between essential objects and background information
- How would we model the relationship between a student and their birthday? A set? A type? A variable? A relation? A function?
- Using  $\text{StudentT}$ ,  $C$  : set of  $\text{StudentT}$  and  $\text{Bday}(s1) = \text{Bday}(s2)$  what is the predicate?

# Students with the same birthday

Consider a class with many students. The instructor bets the students that two or more students have the same birthday. Write a mathematical expression that is True if this condition is met, and False otherwise

- Identify all objects referred to in this problem statement
- Class? Students? Many? Instructor? Birthday?
- Distinguish between essential objects and background information
- How would we model the relationship between a student and their birthday? A set? A type? A variable? A relation? A function?
- Using StudentT,  $C$  : set of StudentT and  $Bday(s1) = Bday(s2)$  what is the predicate?
- $\forall (s1, s2 : \text{StudentT} | s1 \in C \wedge s2 \in C \wedge s1 \neq s2 : Bday(s1) = Bday(s2))$

## Exercise

Consider an array  $D$  with index values ranging from 1 to  $n$ . The subject of this example is part of a specification for a subprogram that will count how many times a particular given value occurs in the array  $D$ .

The goal of this exercise is to write a postcondition for the subprogram, relating the various relevant variables values when the search is complete.



# Exercise Continued

Understand the task in the original language

- Identify objects referred to (look for nouns in the original English text)
  
- Identify missing information

# Exercise Continued

Understand the task in the original language

- Identify objects referred to (look for nouns in the original English text)
- Array  $D$ , index value, times (count), particular given value, relevant variables's value
- Identify missing information

# Exercise Continued

Understand the task in the original language

- Identify objects referred to (look for nouns in the original English text)
- Array  $D$ , index value, times (count), particular given value, relevant variables's value
- Identify missing information
- Names of variable for: index, times (count), particular given value

# Exercise Continued

Understand the task in the original language

- Identify objects referred to (look for nouns in the original English text)
- Array  $D$ , index value, times (count), particular given value, relevant variables's value
- Identify missing information
- Names of variable for: index, times (count), particular given value
- Are there any other relevant variables?

# Exercise

- Identify missing information
- Names of variable for
  - ▶ Index: assume  $i$
  - ▶ Times (count): ask the author of the task, assume *count*
  - ▶ Particular given value: Ask the author of the task, assume *key*
  - ▶ Are there any other relevant variables? (no?)

# Exercise Continued

- Close the gap between English text and mathematics
  - ▶ Reword the English text to be closer to mathematics
  - ▶ Use the English verb *count*

# Exercise Continued

- The English verb *count* means, in programming language and in terms closer to mathematics, *add 1*
- But this is a dynamic (action) concept
- The corresponding static (state, relational) concept in mathematics is the function *addition with 1*, that is  $+1$

## Exercise Continued

- The occurrence of the particular given value in an array element in  $D$
- $D[i] = key$
- A condition for the addition with 1
- The repetition over a variable number of index values suggests quantification with the function addition and with the argument 1
- $+(i : ? | \dots \wedge D[i] = key : 1)$



## Exercise Continued

Identify relationships between essential objects

- Array  $D$ , index value, particular given value:  $D[i] = key$
- Combine the above with count (+ conditionally with 1)
- $count = +(i : \mathbb{N} | \dots \leq i \leq \dots \wedge D[i] = key : 1)$
- Range of  $i$  missing
- Refer to original English text: 1 to  $n$
- $count = +(i : \mathbb{N} | 1 \leq i \leq n \wedge D[i] = key : 1)$

## Exercise: New Glossary Entry

Now we have a new entry for our glossary

- count:  $+(i : \mathbb{N} | \dots \leq i \leq \dots \wedge \dots : 1)$ , where the  $\dots$  defines the range of the quantified variable and the condition for counting

# Modelling Larger Components

- Baber focuses on translating low level mathematical expressions
- In our specification, we will also need to organize the information
- For larger problems, think about your types, Abstract Data Types and Abstract objects
- Interactively switch between types, ADTs, Abstract Objects and detailed mathematical spec
- Example
  - ▶ Modelling game state for [Freecell](#)
  - ▶ What are some potential types?
  - ▶ What are some mathematical expressions you will need?

# Summary

- Knowledge of English and Mathematics is necessary but not sufficient to translate into Mathematics
- Knowledge of subject area also needed
- Translating skills needed
- The three are different

# Summary

- Compile your own glossary
- Make intermediate steps and expressions conscious
- Modularize
- Organize systematically
- KISSS

# Reference

- Baber, Robert L., *Translating English to Mathematics*, 2002