Discrete Math Review

CS 2ME3/SE 2AA4

Justin Staples

Department of Computing and Software
McMaster University

Week of January 22nd, 2018

Outline

1 Motivation
2 Operators
3 Sets
4 Types
5 Quantifiers
6 Practice Exercises
7 References

Motivation

- A software specification that is written in natural language is likely to be interpreted in multiple different ways.
- A formal specification aims to remove these ambiguities.
- Mathematics is the formal language that we use to specify what a program should do.

Unary Operators

- Unary operators take only one operand
- In mathematics, you should already know:
  - $+$ (unary plus)
  - $-$ (unary minus)
  - $\neg$ (logical negation)
- In programming languages, other familiar ones include:
  - ! (logical negation)
  - & (address)
  - $*$ (indirection/pointer)
  - $++ / --$ (increment/decrement)

Binary Operators

- Binary operators take two operands
- For this class, we will primarily use:
  - $+$, $-$, $*$, $/$
  - $\leq$, $\geq$, $<$, $>$, $=$
  - $\vee$, $\wedge$, $\Rightarrow$, $\equiv$
- With so many different operators, it is important to keep in mind the types of the operands and the types of the return values.

Operator Precedence

- Operator precedence gives us a set of rules for determining the order of evaluation when we have expressions with many different operators.
- The rules define a hierarchy of classes. Operators in a higher class will have tighter binding.
- Students that have recently taken 2DM3 will find a good list on the inside cover of the textbook.

Operator Precedence

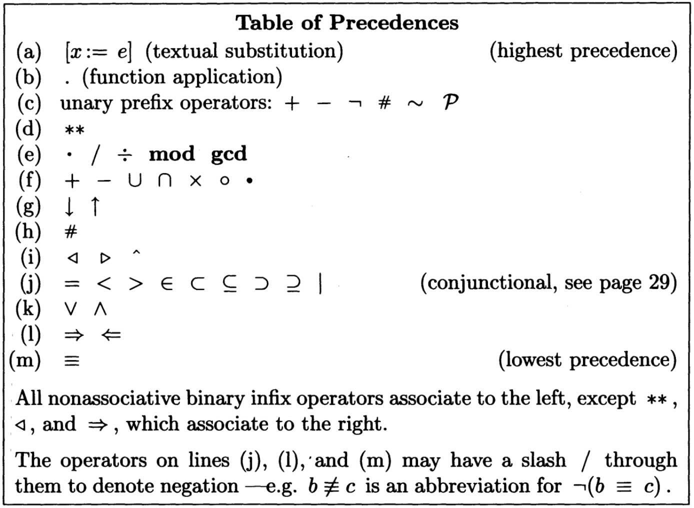

Table of Precedences

(a)	$[x := e]$ (textual substitution)	(highest precedence)
(b)	$.$ (function application)	
(c)	unary prefix operators: $+ \quad - \quad \neg \quad \# \quad \sim \quad \mathcal{P}$	
(d)	$**$	
(e)	$\cdot \quad / \quad \div \quad$ **mod** **gcd**	
(f)	$+ \quad - \quad \cup \quad \cap \quad \times \quad \circ \quad \bullet$	
(g)	$\downarrow \quad \uparrow$	
(h)	$\#$	
(i)	$\triangleleft \quad \triangleright \quad \hat{}$	
(j)	$= \quad < \quad > \quad \in \quad \subset \quad \subseteq \quad \supset \quad \supseteq \quad \mid$	(conjunctional, see page 29)
(k)	$\vee \quad \wedge$	
(l)	$\Rightarrow \quad \Leftarrow$	
(m)	$\equiv$	(lowest precedence)

All nonassociative binary infix operators associate to the left, except $**$, $\triangleleft$, and $\Rightarrow$, which associate to the right.

The operators on lines (j), (l), and (m) may have a slash / through them to denote negation —e.g. $b \not\equiv c$ is an abbreviation for $\neg(b \equiv c)$.

Operator Precedence

- The list shown on the previous slide includes many operators that we will not use, but all of the ones that we will!

Exercise: What is the meaning of the following expression?

$$p \Rightarrow q \wedge r \equiv \neg s \vee t$$

Operator Precedence

- The precedence rules determine where to insert parentheses. Then, the meaning is unambiguous.

Solution:

$$(p \Rightarrow (q \wedge r)) \equiv ((\neg s) \vee t)$$

Operator Associativity

- For operators that are on the same precedence level, the order of evaluation is determined by the associativity rules.
- Operators can be either left associative or right associative.
- Most that you are familiar with are left associative, except for a few exceptions.

Exercise: Consider evaluating 2 ** 3 ** 4 ** 5. Where should the parentheses be inserted? What about $2 + 3 + 4 + 5$?

Operator Associativity

- Exponentiation is right associative!
- Operators that can be evaluated left-to-right and right-to-left are called associative operators. This is the case for $+$, as well as many others.

Solution:

$$2 ** (3 ** (4 ** 5))$$

$$(((2+3)+4)+5)$$

is the same as

$$(2+(3+(4+5)))$$

Sets

- A set is an unordered collection of elements of the same type.
- Any given element is either in the set or it is not.
- Sets do not contain duplicate elements.

Exercise: Identify which of the following are sets

$$A = \{green, blue, yellow\}$$

$$B = \{0, 0, 1, 2\}$$

$$C = \{-5, 5, 10\}$$

$$D = \{\}$$

$$E = \{0, 5, yellow, c\}$$

Sets

Solution:

- A and C are sets because they have unique elements that are all of the same type.
- D is a special set called the empty set.
- B and E are not sets because they either have duplicate elements or elements of different types.

Set Operations

- We can say that an item is included or not included in a set using the member notation: $x \in S$, $y \notin T$.
- The most common set operations include the following:
  - $\cup$ (union)
  - $\cap$ (intersection)
  - $-$ (difference)
  - $\times$ (cartesian product)
  - $\subseteq$ (subset)

Set Building (Enumeration)

- There are two formal ways to define a set. The first is called set enumeration.
- This is essentially a brute force method that lists all the elements in the set.
- This can be good for small sets. For example: $Days = \{Monday, Tuesday, Wednesday, Thursday, Friday, Saturday, Sunday\}$
- What if we wanted the set of all students in the class, or the set of all real numbers?
- For large, or even infinite sets, we need another method.

Set Building (Comrehension)

- Set comprehension provides a way to define a set by stating the properties that the elements of the set must satisfy.
- The general form of the notation is $S = \{x : t \mid R : E\}$, where R is the predicate and E is an expression.
- The predicate is a boolean value that tests which elements to consider.
- The expression is what should be added to the set in the event that the predicate was satisfied.

Exercise: Consider the following set. $P = \{x : \mathbb{N} \mid 1 \leq x < 5 : x^2\}$. Lets use the definition of set comprehension to enumerate the elements in this set.

Set Building (Comrehension)

- The set P contains the square of all the natural numbers from 1 to 4.

Solution:

$$P = \{1, 4, 9, 16\}$$

Types

- Types are used to specify the type of a variable.
- You can think of a type as a set of values. This gives us a nice way of defining types.
- For example, a variable of type colour would be an element of the set $C = \{red, blue, green, yellow, orange, purple\}$.
- A variable of type Integer would be an element of the set $\mathbb{Z} = \{... - 2, 1, 0, 1, 2, ...\}$
- The type of a variable can be specified with the following notation. $x : \mathbb{Z}$ (x is an integer) or $y : C$ (y is a colour).

Primitive Types

- You should already be familiar with a few standard primitive types. These are:
    - Integer ($\mathbb{Z}$)
    - Real ($\mathbb{R}$)
    - Boolean (True, False)
    - Character (the set of ASCII characters)
    - String (a finite sequence of characters)

Custom Types

- We are not limited to the use of these primitives!
- As stated previously, new types can be defined using sets. For example, the colour type mentioned before:
  $C = \{red, blue, green, yellow, orange, purple\}$
- What if we wanted to define a point data type, called PointT?
  PointT = tuple of (x : $\mathbb{R}$, y : $\mathbb{R}$).
- From your A1, the CurveT type might be defined as CurveT = tuple of (x : SeqT, y : SeqT).

Quantifiers

- A quantifier is a shorthand notation for applying an operator many times over. The general form of a quantifier is

$$(*x : X \mid R : P)$$

    - x is an element of type X.
    - R, the range, determines which values of X should be considered.
    - P, the values that the operator $*$ will be applied to.
- This notation looks a lot like the set building notation, and for good reason! Lets see why with an example.

Quantifiers

Exercise: Evaluate p $= (+x : \mathbb{N} \mid 1 \leq x < 5 : x^2)$

Quantifiers

Solution:

- To evaluate, first build the set of elements defined by the range and predicate. For this example, we have already seen this set before (P = {1, 4, 9, 16}).
- Then, apply the operator, in this case $+$, to all of the elements. $p = 1 + 4 + 9 + 16 = 30$

Quantifiers

- We have special names and symbols for quantifications that use $\wedge$ or $\vee$ as the operator.
- For $\wedge$, the universal quantifier, $(\forall x : X \mid R : P)$, asserts that for all x in the range R, the predicate P is satisfied.
- For $\vee$, the existential quantifier, $(\exists x : X \mid R : P)$, asserts that there exists an x in the range R where the predicate P is satisfied.

Practice

Exercise 1: Imagine someone has given you the following description of what it means for a set of integers A to be a subset of set B. 'Set A is a subset of set B if all the integers in A are also in B'. Write this out formally using a quantification. Start with:

$$A \subseteq B \equiv (\mathit{your\ \ answer\ \ here})$$

Practice

Solution:

$$A \subseteq B \equiv (\forall x : \mathbb{Z} \mid x \in A : x \in B)$$

- This can be written out by quantifying over all integers.
- The range tells us to consider only integers that are in set A.
- The predicate returns True when those elements are also in set B.

Practice

Exercise 2: Given the previous definition of subset, determine if A = {1, 2, 3} is a subset of B = {1, 2}.

Solution:

- Using the definition,

$$A \subseteq B \equiv (\forall x : \mathbb{Z} \mid x \in A : x \in B)$$

$$\equiv (1 \in B) \wedge (2 \in B) \wedge (3 \in B)$$

$$\equiv \mathit{True} \wedge \mathit{True} \wedge \mathit{False}$$

$$\equiv \mathit{False}$$

Practice

Exercise 3: Given a set of integers S, what is the meaning of the following expression?

$$(+x : \mathbb{Z} \mid x \in S \wedge x\%2 = 0 : 1)$$

Practice

Solution: the count of all the even integers in S

- $x\%2 = 0$ means that the integer is divisible by 2.
- 1 is contributed every time this occurs, which effectively counts the elements.

Practice

Exercise 4: Evaluate the following quantifications

$$a = (\forall x : \mathbb{Z} \mid -3 <= x <= 3 : (\exists y : \mathbb{Z} \mid -3 <= y <= 3 : x+y = 0))$$

$$b = (\exists x : \mathbb{Z} \mid -3 <= x <= 3 : (\forall y : \mathbb{Z} \mid -3 <= y <= 3 : x+y = 0))$$

Practice

Solution: a = True, b = False

- a is True because for every x in the range $-3 <= x <= 3$, there is a y in the same range that can be added to x to give 0.
- b is false because there is no such number in the range $-3 <= x <= 3$ that gives 0 when added to each other number in that range.
- It is useful to break nested quantifiers up into smaller, simpler parts.
- The order can change the entire meaning.

References

- A Logical Approach to Discrete Math (Gries and Schneider)
- L08 (Lecture Notes 8 Mathematics for MIS)
- The course repository contains a folder of reference material. In particular, the Hoffman and Strooper book gives a good overview of sets, sequences, tuples and types.