

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

Winter 2018

08 Mathematics for MIS (H&S Ch. 3)

Dr. Spencer Smith

Faculty of Engineering, McMaster University

January 23, 2018



08 Mathematics for MIS (H&S Ch. 3)

- Administrative details
- David Parnas
- Review of sets, relations and functions
- Review of logic
- Review of types, sets, sequence and tuples
- Multiple assignment statement
- Conditional rules
- Finite state machines

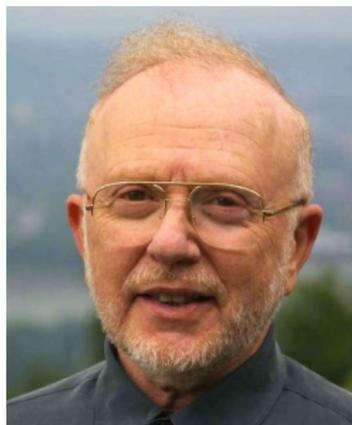
Administrative Details

- Assignment 1
 - ▶ Part 1: January 22, 2018
 - ▶ Partner Files: January 28, 2018
 - ▶ Part 2: January 31, 2018
- Questions on assignment?
- Tutorial this week also includes a mathematics review, with more examples
- [Hoffman and Strooper \(1995\)](#) provides a good review of discrete math for software specification

Avenue Posts (or E-mail) Advice

- First sentence is the question
- Be as clear as possible on what you want
- Provide supporting information - OS, VM?, steps taken, error message
- Consider a screen shot of the error message
- Thank the people that helped you
- For E-mail add identifying information
- For Avenue, consider unchecking “Include original post in reply” checkbox (under Settings)

Who is David Parnas?



- A. A professor emeritus in the Faculty of Engineering
- B. Started the McMaster Software Engineering program
- C. Developed the idea of information hiding in modular design
- D. One of the founding fathers of software engineering
- E. All of the above

Mathematics for Module State Machines

- The material in this lecture should be review
- Shows the simple tools that can be used to build your MIS
- Shows the syntax we will use
- Follows
 - ▶ [Hoffman and Strooper \(1995\)](#), Chapter 3
 - ▶ Gries and Schneider

Sets, Relations and Functions

- A set is an unordered collection of elements
- A binary relation is a set of ordered pairs
- A function is a relation in which each element in the domain appears exactly once as the first component in the ordered pair

Sets

- An element either belongs to a set or it does not
- $x \in S$ versus $x \notin S$
- Defining a set
 - ▶ Enumerate $\{x_1, x_2, x_3, \dots, x_n\}$
 - ▶ Logical condition (rule) from Gries and Schnieder notation $\{x : t \mid R : E\}$, where R is a predicate and E is an expression
 - ▶ An integer range $[2..4] = \{2, 3, 4\}$, $[7..4] = \{\}$
- Examples
 - ▶ $S = \{1, 7, 6\}$
 - ▶ $S = \{x : \mathbb{Z} \mid 1 \leq x \leq 4 : x\}$
 - ▶ $S = \{x : \mathbb{N} \mid 1 \leq 100 : x^2\}$
- Does $\{1, 7, 6\} = \{7, 1, 6\}$?

Sets

How would you write the set of points C inside a circle centered at the origin with a radius of r ?

- What is the type of each element in the set?
- What is the dummy variable?
- What is the range?

Sets

How would you write the set of points C inside a circle centered at the origin with a radius of r ?

$$C(r) = \{x, y : \mathbb{R} \mid x^2 + y^2 \leq r^2 : \langle x, y \rangle\}$$

- What if the circle is centered at (x_c, y_c) ?

Relations

- Let $\langle x, y \rangle$ denote an ordered pair
 - ▶ $dom(R) = \{x | \langle x, y \rangle \in R\}$
 - ▶ $ran(R) = \{y | \langle x, y \rangle \in R\}$
- Defining a relation
 - ▶ Enumerate $\{\langle 0, 1 \rangle, \langle 0, 2 \rangle, \langle 2, 3 \rangle\}$
 - ▶ Rule $\{x, y : \mathbb{Z} | x < y : \langle x, y \rangle\}$

Functions

- Let $\langle x, y \rangle$ denote an ordered pair
- Each element of the domain is associated with a unique element of the range
- Defining a function
 - ▶ Enumerate $\{\langle 0, 1 \rangle, \langle 1, 2 \rangle, \langle 2, 3 \rangle\}$
 - ▶ Rule $\{x, y : \mathbb{Z} \mid y = x^2 : \langle x, y \rangle\}$
- Notation
 - ▶ $f(a) = b$ means $\langle a, b \rangle \in f$
 - ▶ $f(x) = x^2$
 - ▶ $f : T_1 \rightarrow T_2$
 - ▶ $\{x_1, x_2, y : \mathbb{Z} \mid y = x_1 + x_2 : \langle \langle x_1, x_2 \rangle, y \rangle\}$
- Is $\{\langle 0, 1 \rangle, \langle 0, 2 \rangle, \langle 2, 3 \rangle\}$ a function?
- Is $\{x, y : \mathbb{Z} \mid y^2 = x : \langle x, y \rangle\}$ a function?

Logic

- A logical expression is a statement whose truth values can be determined ($6 < 7?$)
- Truth values are either *true* or *false*
- Complex expressions are formed from simpler ones using logical connectives ($\neg, \wedge, \vee, \rightarrow, \leftrightarrow$)
- Truth tables
- Evaluation
 - ▶ Decreasing order of precedence: $\neg, \wedge, \vee, \rightarrow, \leftrightarrow$
 - ▶ Evaluate from left to right
 - ▶ Use rules of boolean algebra

Quantifiers

- Variables are often used inside logical expressions
- Variables have types
- A type is a set of values from which the variable can take its value
- Often quantify a logical expression over a given variable
 - ▶ Universal quantification
 - ▶ Existential quantification

Quantifiers Continued

- Bound variables appear in the scope of the quantifier
- Free variables are not bound to any quantifier
- Free variables in an expression often mean that we cannot determine the truth value of the expression
- Gries and Schneider use the following notation $(\forall x : t | R : P)$ where R is a predicate and P is a predicate
- \forall is applied to the values P for all x in t for which range R is true

Types, Sets, Sequence and Tuples

- A type is a set of values, so any precisely defined set is a type
- Primitive types are integer, boolean, character, string and real
- User-defined types
 - ▶ The set of values has to be given
 - ▶ Often use type constructors
- Useful type constructors
 - ▶ Set
 - ▶ Sequence
 - ▶ Tuple

Types

- Specify the type of a variable
 - ▶ $x_1, x_2, \dots, x_n : T$
 - ▶ $x : \textit{integer}$ or $x : \mathbb{Z}$
 - ▶ $a, b, c : \textit{string}$
- Type definition
 - ▶ $T = d$
 - ▶ $\textit{float} = \textit{real}$
 - ▶ $\textit{colour} = \{\textit{red}, \textit{white}, \textit{blue}\}$
 - ▶ $\textit{testtype} = \{\textit{uniaxial}, \textit{biaxial}, \textit{shear}\}$
 - ▶ $x : \textit{testtype}$
 - ▶ $\textit{motion}T = \{\textit{forward}, \textit{backward}, \textit{stop}\}$

Primitive Types

- Integer

- ▶ $\{\dots - 2, -1, 0, 1, 2, \dots\}$
- ▶ $+, -, \times, /$
- ▶ $=, \neq$
- ▶ $<, \leq, \geq, >$

- Real

- ▶ $\{\textit{all real numbers}\}$
- ▶ $+, -, \times, /, \sin(), \cos(), \exp()$ etc.
- ▶ $=, \neq$
- ▶ $<, \leq, \geq, >$

Primitive Types Continued

- Boolean type
 - ▶ $\{true, false\}$
 - ▶ $\neg, \wedge, \vee, \rightarrow, \leftrightarrow$
- Char type
 - ▶ Set of ASCII characters
 - ▶ Character values appear in quotes $'a', 'b', 'c'$, etc.
 - ▶ $=, \neq$

Primitive Types Continued

- String type
 - ▶ All finite sequences of characters
 - ▶ String constants are in double quotes `"abc"`
 - ▶ $s[i..j]$ is the substring of s from position i to position j
 - ▶ $s_1||s_2$ concatenates strings s_1 and s_2
 - ▶ $=, \neq$ for is equal and not equal
 - ▶ \in, \notin for is member and not a member
 - ▶ $s[i]$ is the i th character of s
 - ▶ $|s|$ is the length of s
 - ▶ Positions in strings are zero relative

Sets

- A set is an unordered collection of elements of the same type
- Declare a set of type T as *set of T*
- Example
 - ▶ $T = \text{set of } \{\text{red}, \text{green}, \text{blue}\}$ defines type T as the power set of $\{\text{red}, \text{green}, \text{blue}\}$
 - ▶ $x : \text{set of integer}$
- What are some possible values for $x : \text{set of integer}$?

Operations on Sets

- \cup union
- \cap intersection
- $-$ difference
- \times Cartesian product
- $=, \neq$ equal, not equal
- \in, \notin member, non-member
- $|s|$ size of set s

Sequences

- A sequence is an ordered collection of elements of the same type
 - ▶ Elements can occur more than once
 - ▶ Sometimes referred to as a list
 - ▶ Similar to an array
- Declare a sequence of type T by *sequence of T*
- $\langle x_0, x_1, \dots, x_n \rangle$ for $n \geq 0$ for a sequence with elements x_0, x_1, \dots, x_n
- $\langle \rangle$ is the empty sequence
- Position in a sequence is zero relative

Sequences Continued

- Examples
 - ▶ $T = \text{sequence of } \{red, green, blue\}$ defines the type T as the set of all sequences of elements from $\{red, green, blue\}$
 - ▶ $x : \text{sequence of integer}$
- Fixed-length sequence of type T with length l
 - ▶ $\text{sequence } [l] \text{ of } T$
 - ▶ l is a positive integer
 - ▶ $\text{sequence } [l_1, l_2, \dots, l_n] \text{ of } T$ is a shorthand for $\text{sequence } [l_1] \text{ of sequence } [l_2] \text{ of } \dots \text{sequence } [l_n] \text{ of } T$

Operations on Sequences

- $s[i..j]$ is the subsequence of s from position i to position j
- $s[i..j]$ is undefined if $i \notin [0..|s| - 1] \vee j \notin [0..|s| - 1]$
- $s_1 || s_2$ concatenates sequences s_1 and s_2
- $=, \neq$ for is equal and not equal
- \in, \notin for is member and not a member
- $s[i]$ is the i th element of s
- $s[i]$ is undefined if $i \notin [0..|s| - 1]$
- $|s|$ is the length of s
- A string is a sequence of characters

Tuples

- A tuple is a collection of elements of possibly different types
- Each tuple has one or more fields
- Each field has a unique identifier called the field name
- Similar to a record or a structure
- To declare a tuple use
 - ▶ *tuple of* $(f_1 : T_1, f_2 : T_2, \dots, f_n : T_n)$ with $n \geq 1$
 - ▶ f_i is the name of the i th field
 - ▶ T_i is the type of the i th field
 - ▶ *tuple of* $(f_1, f_2, \dots, f_n : T)$ if all fields are of the same type

Example Tuples

- Examples
 - ▶ pair = tuple of (id: integer, val: string)
 - ▶ experimentT = tuple of (b_{cond} : bcondT, control: controlT)
 - ▶ pointT = tuple of (x: real, y: real)
- Define the value of a tuple by using an expression of the form
 - ▶ $\langle x_1, x_2, \dots, x_n \rangle$
 - ▶ $\langle 4, "cat" \rangle$ is a value of type pair

Operations on Tuples

- $=, \neq$ equal, not equal
- $t.f$ is the value of field f of tuple t

Using Type Constructors

- $bcondT = \{uniaxial, biaxial, multiaxial, shear\}$
- $controlT = \{load_controlled, displacement_controlled\}$
- $experimentT = \text{tuple of } (b_{cond} : bcondT, control : controlT)$
- $experiment : experimentT$
- $directionT = \{clockwise, counterclockwise\}$
- $powerT = [MIN_POWER...MAX_POWER]$
- $motorT = \text{tuple of } (powerOn : Boolean, direction : directionT, powerLevel : powerT)$

Multiple Assignment Statement

- $v_1, v_2, \dots, v_n := e_1, e_2, \dots, e_n$ with $n \geq 1$
- The v_i s are distinct variables and each e_i is an expression of the same type as v_i
- Compute the values of all the expression e_i and then assign these values simultaneously
- Example
 - ▶ $x, y := 0, 10$
 - ▶ $x, y := 10, x$
 - ▶ $x, y := y, x$
- Convenient for defining the meaning of pieces of code
- Use as a function on the state space of a program

Uses of Conditional Rules

- To define the value of a function
- $\min(x, y) = (x \leq y \Rightarrow x \mid x > y \Rightarrow y)$
- To define the meaning of a program
 - ▶ If $(x < y)$ then $z := x$ else $z := y$
 - ▶ $(x < y \Rightarrow z := x \mid x \geq y \Rightarrow z := y)$
 - ▶ $(x < y \Rightarrow x, y := x, y \mid x \geq y \Rightarrow x, y := y, x)$
- Conditional rules can be expressed in tables

Finite State Machines

- A FSM is a tuple $(S, s_0, I, O_E, O_C, T, E, C)$ where
- S is a finite set of states
- s_0 is the initial state in S ($s_0 \in S$)
- I is a finite set of inputs
- $T : S \times I \rightarrow S$ is the transition function
- O_E is a finite set of event outputs
- $E : S \times I \rightarrow O_E$ is the event output
- O_C is a finite set of condition outputs
- $C : S \rightarrow O_C$ is the condition output

Homework Answer for `intersect(c)`

out : =?

- What is the type of out?
- How would you calculate the output (in words)?
- `intersect(c)` knows the centre and radius of the current object – how do we get the centre and radius of `c`?
- What is the type of the set of points inside a circle?

CircleT ADT MIS: Syntax

Access Routine Syntax

Routine name	Inputs	Outputs	Exceptions
new CircleT	real, real, real	CircleT	
xc		real	
yc		real	
r		real	
area		real	
...
intersect	CircleT	Boolean	

State variables

xc: real

yc: real

r: real

Homework Answer for intersect(c)

intersect(c):

out :=

$$\text{circlePts}(xc, yc, r) \cap \text{circlePts}(c.xc, c.yc, c.r) \neq \emptyset$$

circlePts(xc, yc, r): $\mathbb{R} \times \mathbb{R} \times \mathbb{R} \rightarrow$ set of \mathbb{R}^2

circlePts(xc, yc, r)

$$\equiv \{x, y : \mathbb{R} \mid (x - xc)^2 + (y - yc)^2 \leq r^2 : \langle x, y \rangle\}$$

Homework Answer for intersect(c)

out :=

$$(\exists \langle x, y \rangle : \mathbb{R}^2 \mid (x - xc)^2 + (y - yc)^2 \leq r^2 : \\ (x - c.xc)^2 + (y - c.yc)^2 \leq (c.r)^2)$$

$$\text{out} := \text{dist}(xc, yc, c.xc, c.yc) \leq (r + c.r)$$

where

$$\text{dist}(x_1, y_1, x_2, y_2): \mathbb{R} \times \mathbb{R} \times \mathbb{R} \times \mathbb{R} \rightarrow \mathbb{R}$$

$$\text{dist}(x): \equiv \sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2}$$

Which version is most abstract?

Which version is closest to code?

References

- Hoffman and Strooper, *Software Design, Automated Testing and Maintenance*, International Thomson Computer Press, 1995.
- Gries and Schneider, *A Logical Approach to Discrete Math*, Springer, 1993.