What is Symbolic Computing

Thank you to Prof. Roosen-Runge
for the slides on
background and history.

What is Symbolic Computing

- **Computing on**
  - **non-numbers**
  - **non-character-string**
  - **use atoms instead of numbers and strings**
- **Building structures from atoms**
  - **lists, trees, terms, clauses, propositions, etc.**

Symbols are Used to Describe

- **Symbolic programming**
  - **programming that uses descriptions and creates descriptions**
- **Reflexive application of symbolic programming**
  - **compute a program from a description**
  - **often used to create special interactive programming environments (IDEs)**

Operational Programming

- **Basic, Pascal, C, Java, etc.**
  - **require describing how something is computed**
  - **program describes a sequence of operations.**
  - **not describing what is computed**

```
j ← 1
while j ≤ max {
    print item(j)
    j ← j + 1
}
```

Denotational programming

- **Describes what to compute**

```
(apply  print) : item
```

  - **Denotational program has a mathematical meaning**
    - » **uses mathematical objects such as functions, relations, etc.**
  - **Program or segment of a program denotes or names that object.**

Denotation & Logic

- **Denotational program describes its result in terms of logical properties and relationships.**
- **Examples of denotational languages:**
  - **Lisp**
  - **Prolog**
  - **APL**
  - **ML**

Timeless vs. State-change

- **Denotational semantics uses mathematical language**
  - **Timeless propositions**
  - **Nothing changes**
    - » **x = x + 1 is false**
- **Operational semantics uses**
  - **language of states (memory) and change-of-state**
    - » **x ← x + 1 describes a change in state of x**
    - » **= in C/C++,Java, and Fortran**
    - » **:= in Pascal and Eiffel**

What is a Denotation?

- **Denotation = object described by an expression or referred to by a name.**
- **In denotational programming languages, the object is mathematical**
  - **number**
  - **abstract symbol**
  - **function**
  - **equation or proposition**

Denotation History

- **Concept of denotation comes from the theories of how logic connects to mathematics worked out by Bertrand Russell & Albert North Whitehead at the turn of the 20'th century (famous book: Principia Mathematica)**

- **Based on ideas from German logician Gottlob Frege**

  Invented the concepts of the predicate calculus and quantifiers: (for all, there exists)

Description and Prescription

Programs are both descriptions and prescriptions

x = y + 3

- **interpreted operationally (prescription)**
  program ≡ instructions to underlying machine as to what to do

  Add 3 to y and store result in x

- **Interpreted denotationally (description)**
  program ≡ description of mathematical relationship between input and output.

  When executed, value of x equals value of y + 3

Prescription Example

```
palindrome ( String x ) : boolean  is
   int half ← x.length div 2
   for  i : 0 .. half  do
         if  x.charAt ( i ) ≠   x.charAt ( x.length – 1– i ) )
         then return false
         fi
   end for
   return true
end palindrome
```

Description Example

```
// Given the following two functions.
   // Result =  (x = y)
       match ( String x, String y ) : boolean

   // Result is the string reversal of x.
       reverse ( String x ) : String

// Then ...

   palindrome ( String x ) : boolean is
       return match ( x, reverse (x) )
   end palindrome
```

Functional vs. Declarative

- **Functional (Lisp-like)**
  - palindrome (x) is x = rev (x))
    where
    - » rev (nil) is nil -- reversal of empty is empty
    - » and rev (w ^ x) is append (rev (x), w)
- **Declarative (Prolog-like)**
  - palindrome (x) if rev (x, x)
    where
    - » rev ([], []) -- Empty is the reversal of empty
    - » and rev (w ^ x, y) if rev (x, z)
      and append (z, w , y)

Denotational Semantics

- **Can languages like C, Java be given a denotational semantics?**
- **Yes, but the result is very complicated.**
  - **The denotations (mathematical objects) have to model the computer's memory and changes of state.**
  - **This is taken up in greater detail in CSE 3341.**

In a Nutshell

- **We investigate symbolic computation by looking at programming which**
  - **manipulates symbols rather than just characters and numbers**
  - **uses symbolic descriptions to specify what is to be computed, rather than how to compute**

General Goals

- **Understand important ideas and historical context in computer science**
- **Extend understanding of programming concepts and vocabulary**
- **Learn to adapt to a new mindsets – actually two new mindsets!**
  - **pure functional programming – Lisp**
  - **declarative programming – Prolog**