Algorithmic Thinking and Programming

Many concepts connected with computing are valuable not only when dealing with computers but in everyday situations as well. Algorithmic thinking is one such concept.

Algorithm

- An algorithm is a procedural or systematic method for solving a problem.
- An algorithm has five basic properties [Knuth]...
  - Finiteness -- the process completes after a finite number of steps
  - Definiteness -- each step must be precisely defined
  - Input -- the data the algorithm processes
  - Output -- the result, or indication that the result is not found
  - Effectiveness -- the steps must be sufficiently primitive that they can be performed by the executing agent

- When one specifies an algorithm, one is programming … a program is an instance of an algorithm.
Ex: Directions To Northgate Cinema

- When we give directions, we are giving a process for reaching a destination
  - To go to the Northgate Cinema, exit I-5 at Northgate, go straight through the light, take a left once in the mall parking lot and go one block.
- Weaknesses of these directions ...
  - Finiteness: It is finite, at least under some circumstances
  - Definiteness: Assumes one is traveling on I-5, but doesn’t say which direction; or equivalently what is Northgate’s exit number; assumes a particular exit, though there are two
  - Input: Should have the starting location as input
  - Output: Ill-specified -- directions lead to back of cinema
  - Effectiveness: Is vague about navigating in mall parking lot

Better Directions

- The goal in giving directions is to make them doable (effective) and unambiguous (definite), attending to details such as the starting point (input) and the ending point (output)
- Can we develop better directions?
Key Features of Algorithms/Programs

A few primitive ideas are used in all algorithms …

- Assignment -- associating a name with a value
- Conditional operations -- mechanisms for making decisions based on input or computed data that determine the next steps in the computation
- Repetition operations -- mechanisms for repeatedly performing certain steps in the computation, that assure termination (finiteness) and the ability to reference different data on different repetitions
- Functional abstraction -- a mechanism for encapsulating the steps of a commonly used operation to become a basic unit of computation
- Functional decomposition -- a process of breaking a complex task into simpler steps

Assignment

- Assignment is associating a name with a value … it’s like naming quantities in algebra, but slightly different
- In programming we write …

  \[ \text{interest\_rate} = 5.5 \]

  - The letters “interest\_rate” are called a variable in programming because they can take on many values
  - Variables have to be single words, always beginning with a letter, so the underscore symbol (_) connects the parts for readability … capitals work for this purpose too:
  - Example variables: \textbf{interestRate}, \textit{x15}, \texttt{b44zrptq261}
  - The equal sign is read: “is assigned,” “becomes” or “gets”
  - A good metaphor for a variable is a labeled container … since we speak of the “contents” of a variable, i.e. its current value

\[ \text{interest\_rate} \quad 5.5 \]
Conditionals

- Conditional operations allow variables to be tested to determine what the next operation should be.
- The If-Then-Else is the most widely used conditional.
- Structurally, the If-Then-Else has the form:
  
  ```
  If <condition goes here> Then
  <Then-clause instructions go here>
  Else
  <Else-clause instructions go here>
  End If
  ```

  Parts of a conditional:
  - Key words: If, Then, Else, End If
  - Test or Predicate, between If and Then
  - Then clause
  - Else clause (optional)
  - End If terminator

A Conditional Example For VB6

- The If-Then-Else used to change the month ...
  ```
  Next_week = Today + 7
  If Next_week > 30 Then
  Month = "May"
  Next_week = Next_week - 30
  Else
  Month = "April"
  End If
  ```

- An alternative ...
  ```
  Next_week = Today + 7
  Month = "April"
  If Next_week > 30 Then
  Month = "May"
  Next_week = Next_week - 30
  End If
  ```

- Another alternative ...
  ```
  Next_week = Today + 7
  Month = "April"
  If Next_week > 30 Then Month = "May"
  If Next_week > 30 Then Next_week = Next_week - 30
  ```
Program & Algorithm Differences

We will return to this point later

- A program is an instance of an algorithm, meaning that the algorithm is a more abstract concept of a process than is the program.
- The program has been created with a particular set of properties, specific representations, input assumptions, etc., but implementing the underlying logic of the algorithm.
- A different set of conditions leads to a different program implementing the algorithm’s logic.

Fundamental principle: There can be different instances of a single abstract idea … algorithms and programs illustrate the principle.