Computational Problem Solving

In computational problem solving, a **representation** is needed of the *relevant* aspects of a given problem to solve.

The means of solving the problem is based on **algorithms** that operate within the given representation.

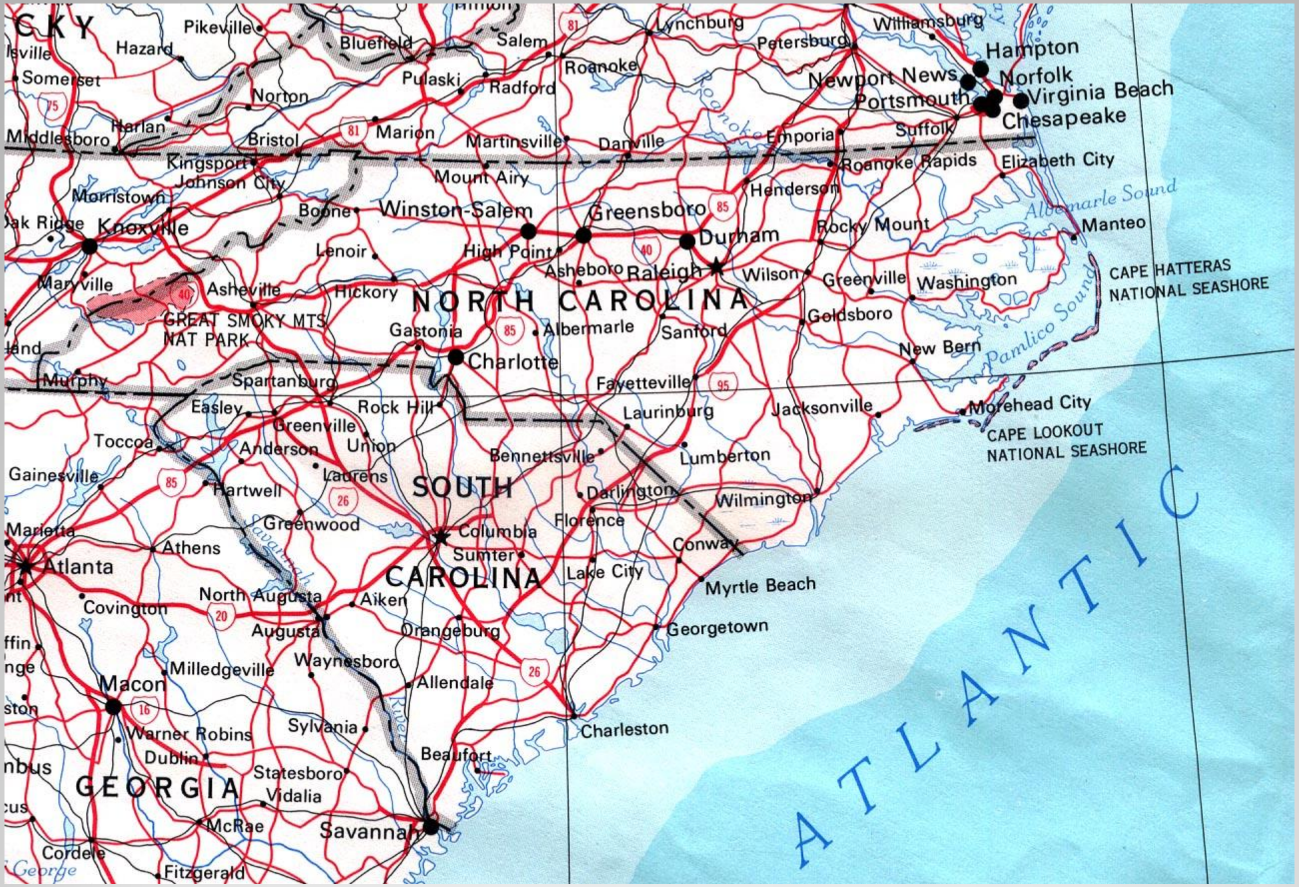
A representation that leaves out details is a form of **abstraction**.

Thus, computational problem solving fundamentally involves the development and use of “**executable abstractions.**”

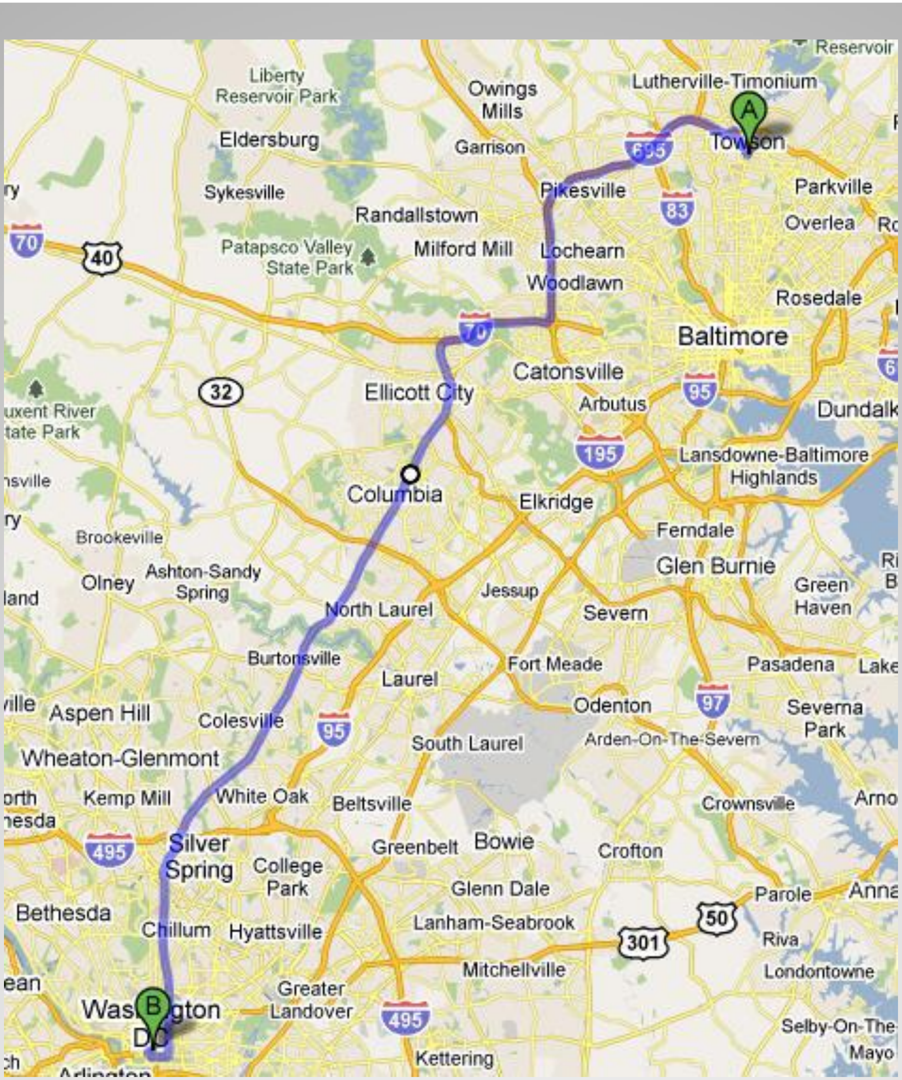
A **computer program** is an executable abstraction. It consists of both a representation of a problem, and implementation of the needed algorithms.

A road map is an abstraction since it does not denote *every* aspect of the terrain. However, it does represent what is needed to be able to:

- Find out how to get somewhere
- Find the direct distance between two points (by use of the map scale)


CKY
Pikeville
Hazard
Bluefield
Salem
Lynchburg
Williamsburg
Petersburg
Hampton
Norfolk
Newport News
Portsmouth
Virginia Beach
Chesapeake
Somerset
Pulaski
Radford
Roanoke
Norton
Harlan
Middlesboro
Bristol
Marion
Martinsville
Danville
Emporia
Suffolk
Kingsport
Johnson City
Mount Airy
Roanoke Rapids
Elizabeth City
Morristown
Henderson
Albemarle Sound
Boone
Winston-Salem
Greensboro
Durham
Rocky Mount
Manteo
Knoxville
Oak Ridge
Lenoir
High Point
Asheboro
Raleigh
Wilson
Greenville
Washington
CAPE HATTERAS
NATIONAL SEASHORE
Maryville
Asheville
Hickory
NORTH CAROLINA
GREAT SMOKY MTS
NAT PARK
Gastonia
Albermarle
Sanford
Goldsboro
Pamlico Sound
New Bern
Charlotte
Murphy
Spartanburg
Fayetteville
Easley
Rock Hill
Laurinburg
Jacksonville
Morehead City
Greenville
Toccoa
Anderson
Union
CAPE LOOKOUT
NATIONAL SEASHORE
Bennettsville
Lumberton
Laurens
Gainesville
SOUTH
Hartwell
Darlington
Wilmington
Greenwood
Florence
Columbia
Conway
Marietta
Athens
Atlanta
Sumter
Lake City
CAROLINA
North Augusta
Aiken
Myrtle Beach
Covington
Augusta
Orangeburg
Georgetown
Milledgeville
Waynesboro
Macon
Allendale
Warner Robins
Sylvania
Charleston
Dublin
Beaufort
GEORGIA
Statesboro
Vidalia
McRae
Savannah
Cordele
Fitzgerald
Savannah River
Roanoke
ATLANTIC
81
75
85
40
95
26
20
16

Google maps use a representation of roadways, and an algorithm for determining the best routes of travel from one location to another.

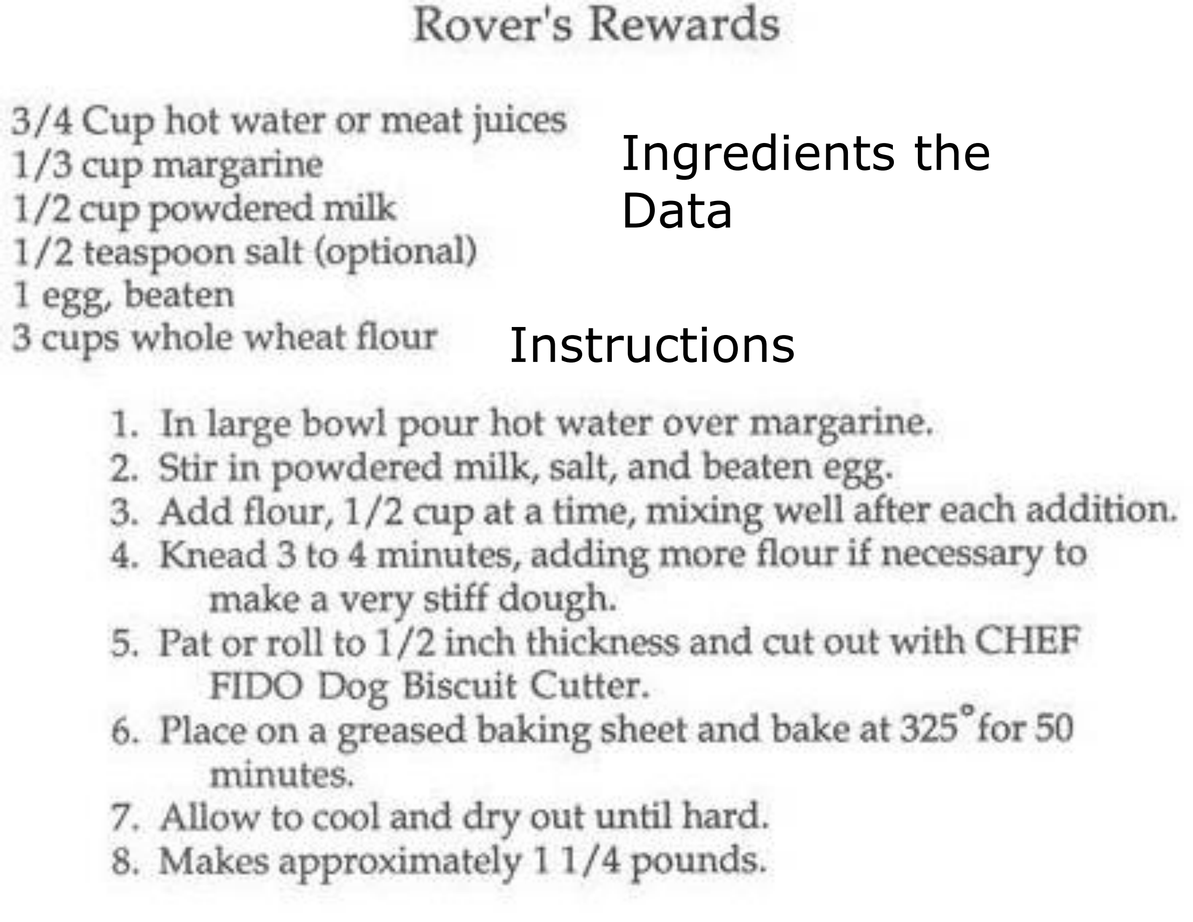

Reservoir
Liberty Reservoir Park
Owings Mills
Lutherville-Timonium
A
Towson
Eldersburg
Garrison
695
Sykesville
Pikesville
Parkville
Randallstown
83
Overlea
70
40
Patapsco Valley State Park
Milford Mill
Lochearn
Woodlawn
Rosedale
70
Baltimore
Catonsville
32
Ellicott City
95
Arbutus
Dundalk
Patuxent River State Park
195
Lansdowne-Baltimore Highlands
Columbia
Elkridge
Ferndale
Brookeville
Glen Burnie
Ashton-Sandy Spring
Olney
Jessup
Green Haven
North Laurel
Severn
Burtonsville
Fort Meade
Pasadena
Laurel
Aspen Hill
Colesville
95
Odenton
97
Severna Park
South Laurel
Arden-On-The-Severn
Wheaton-Glenmont
Kemp Mill
White Oak
Beltsville
Crownsville
495
Silver Spring
College Park
Greenbelt
Bowie
Crofton
Bethesda
Glenn Dale
Parole
50
Chillum
Hyattsville
Lanham-Seabrook
301
Riva
Mitchellville
Londontowne
Washington DC
B
Greater Landover
495
Selby-On-The
Mayo
Kettering
Arlington

What is an algorithm?

An **algorithm** is a step-by-step method for solving a certain type of problem, such as sorting a list. All algorithms eventually terminate and give a solution to a particular problem instance.

EVERYDAY EXAMPLE: Cookbook recipe

A Cookbook Recipe is an Algorithm



Rover's Rewards

3/4 Cup hot water or meat juices
1/3 cup margarine
1/2 cup powdered milk
1/2 teaspoon salt (optional)
1 egg, beaten
3 cups whole wheat flour

Ingredients the Data

Instructions

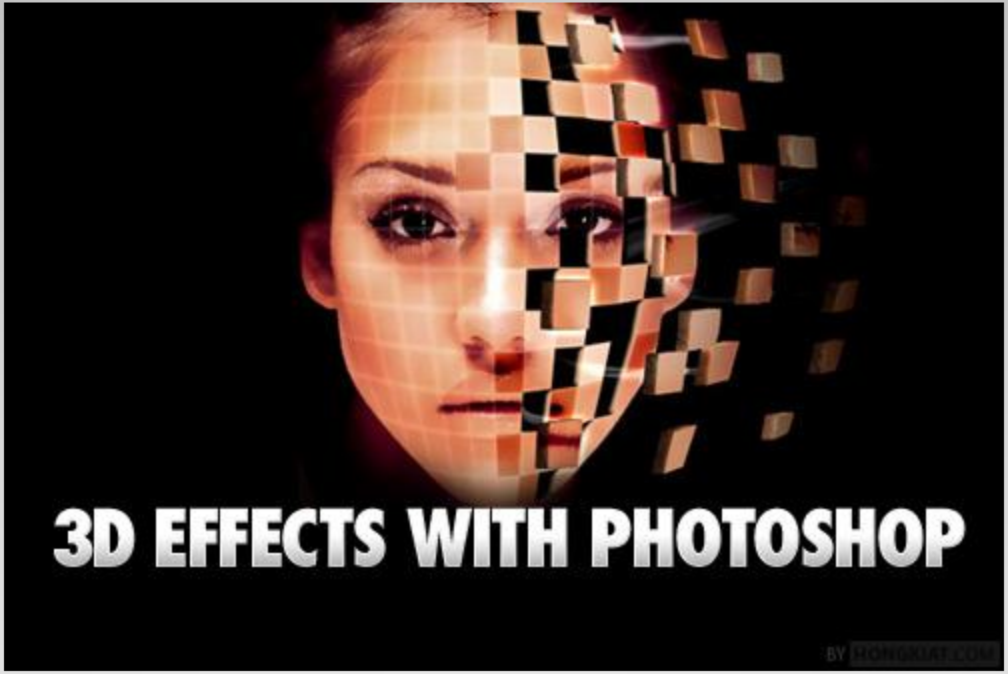
1. In large bowl pour hot water over margarine.
2. Stir in powdered milk, salt, and beaten egg.
3. Add flour, 1/2 cup at a time, mixing well after each addition.
4. Knead 3 to 4 minutes, adding more flour if necessary to make a very stiff dough.
5. Pat or roll to 1/2 inch thickness and cut out with CHEF FIDO Dog Biscuit Cutter.
6. Place on a greased baking sheet and bake at 325° for 50 minutes.
7. Allow to cool and dry out until hard.
8. Makes approximately 1 1/4 pounds.

Photoshop: A Bundle of Algorithms

Photoshop is a good example of the use of algorithms.

The program is essentially a collection algorithms that can affect a digital image in ways that used to be done physically, for example:

- Adjust the colors
- Adjust the contrast
- Special effects


3D EFFECTS WITH PHOTOSHOP
BY HONGKIAT.COM

Example Algorithms

- Instructions for putting together furniture
- Instructions on lid of a washing machine
- Instructions for performing a magic trick
- Euclid’s Algorithm

Euclid’s Algorithm

Euclid’s Algorithm for Finding the Greatest Common Divisor of Two Numbers

Step 1: Assign M the value of the larger of the two input values.

Step 2: Divide M by N, can call the remainder R.

Step 3: If R is not 0, then assign M the value of N, assign N the value of R, and go to step 3; otherwise, the greatest common divisor is N.

The Fundamental Components of an Algorithm

Algorithms contain only a few types of fundamental instructions, that in combination, can be used to compute everything that can be computed.

What CAN be computed is a very fundamental question in computer science.

Three Fundamental Forms of Control

- Sequential
  Instructions executed in order

- Selection
  Which instructions executed based a condition

- Repetition (iteration / loops)
  A set of instructions repeatedly executed while a given condition is true

With these three forms of control, can compute anything that can be computed

Algorithms and Computers: A Perfect Match

Computers have the property of being able execute a series of instructions reliably and very quickly.

If we can figure an algorithm to solve a given type of problem, then all instances of that problem can be automatically solved by computers.

Basic Program Steps

- Input of values
- Variables to store values in
- Use of variables in calculations
- Control of order of instructions executed
- Output of results

Farhenheit to Celsius Example

```
print ‘Convert Celsius to Fahrenheit'
celsius = input('Enter degrees Celsius: ')
fahrenheit = (9.0/5.0 * celsius) + 32
print fahrenheit, ‘degrees Fahrenheit‘
```

Program Output

```
Degrees Celsius to degrees Fahrenheit
Enter degrees Celsius: 100
212.0 degrees Fahrenheit
```

Variables Used

```
print ‘Convert Celsius to Fahrenheit'
celsius = input('Enter degrees Celsius: ')
fahrenheit = (9.0/5.0 * celsius) + 32
print fahrenheit, ‘degrees Fahrenheit‘
```

Variables on left side of = means ASSIGNMENT

Variables anywhere else means USE of its current value.

Variable Assignment

The instruction,

```
n = 10
```

Does not mean that n equals 10, but rather that the value 10 is **assigned** to variable n.

```
n ← 10
```

Variable Use

When a variable appears within an expression, its CURRENT VALUE is used.

In the following, the expression on the right of the assignment is first evaluated, and then the result assigned as the NEW value of `n`:

```
n = 10
n = n + 1
```

Result is that variable n will hold the value 11.

Program Output (Greeting)

```
# Temperature Conversion (Celsius-Fahrenheit)

print ‘Degrees Celsius to degrees Fahrenheit'

celsius = input('Enter degrees Celsius: ')
fahrenheit = (9.0/5.0 * celsius) + 32

Print fahrenheit, 'degrees Fahrenheit'
```

Program Input (Celsius)

```
# Temperature Conversion (Celsius-Fahrenheit)

print ‘Degrees Celsius to degrees Fahrenheit'

celsius = input('Enter degrees Celsius: ')
fahrenheit = (9.0/5.0 * celsius) + 32

print fahrenheit, 'degrees Fahrenheit'
```

Variable Assignment (celsius)

```
# Temperature Conversion (Celsius-Fahrenheit)

print ‘Degrees Celsius to degrees Fahrenheit'

celsius = input('Enter degrees Celsius: ')
fahrenheit = (9.0/5.0 * celsius) + 32

print fahrenheit, 'degrees Fahrenheit'
```

Expression Evaluation

```
# Temperature Conversion (Celsius-Fahrenheit)

print ‘Degrees Celsius to degrees Fahrenheit'

celsius = input('Enter degrees Celsius: ')
fahrenheit = (9.0/5.0 * celsius) + 32

print fahrenheit, 'degrees Fahrenheit'
```

NOTE: * is the symbol for multiplication in most programming languages

Variable Assignment (fahrenheit)

```
# Temperature Conversion (Celsius-Fahrenheit)

print ‘Degrees Celsius to degrees Fahrenheit'

celsius = input('Enter degrees Celsius: ')
fahrenheit = (9.0/5.0 * celsius) + 32

print celsius, 'degrees Celsius equals',
   fahrenheit, 'degrees Fahrenheit'
```

Program Output (Celsius)

```
# Temperature Conversion (Celsius-Fahrenheit)

print ‘Degrees Celsius to degrees Fahrenheit'

celsius = input('Enter degrees Celsius: ')
fahrenheit = (9.0/5.0 * celsius) + 32

print fahrenheit, 'degrees Fahrenheit'
```