

To Ponder

Computer Science and Engineering @ The Ohio State University

Does a problem get *easier* or *harder* to solve if I give you *less* information?

Computer Science & Engineering

Computer Science and Engineering @ College of Engineering @ The Ohio State University

An Introduction
(and some advanced concepts too!)

Prof. Paul Sivilotti
sivilotti.1@osu.edu

Where is Engineering?

Computer Science and Engineering @ The Ohio State University



Where is Computer Science?

Computer Science and Engineering @ The Ohio State University



Computer Science is Also...

Computer Science and Engineering @ The Ohio State University



The First Computer Scientist

Computer Science and Engineering @ The Ohio State University



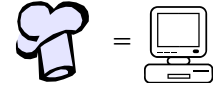
Ada Byron King,
Countess of Lovelace
1815-1852

Computers and Programs

- Computer: a device that “computes”
 - Takes **inputs**, produces **output**
- Program: a sequence of instructions
 - **How** to produce the output
- Contrast
 - Computers: smaller, faster, cheaper
 - Programs: larger and more complicated!

Now We’re Cooking!

- Chef = computer
- Recipe = program



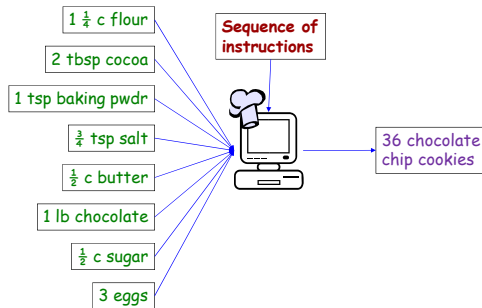
1. Preheat oven to 350°
2. Sift together flour, cocoa, baking powder, salt
3. Melt 1/2c butter and 1lb chocolate
4. Stir 1/2c sugar into chocolate mixture
5. Stir in 3 large eggs
6. Stir in dry ingredients
7. Add chocolate chunks
8. Form into rounded balls (1T each)
9. Bake 10 min

```

1 // Action: getIngredients() method.
2 // Action: 2.0 // Action: getIngredients()
3 // Action: 3.0 // Action: getIngredients()
4 // Action: 4.0 // Action: getIngredients()
5 // Action: 5.0 // Action: getIngredients()
6 // Action: 6.0 // Action: getIngredients()
7 // Action: 7.0 // Action: getIngredients()
8 // Action: 8.0 // Action: getIngredients()
9 // Action: 9.0 // Action: getIngredients()
10 // Action: 10.0 // Action: getIngredients()
11 // Action: 11.0 // Action: getIngredients()
12 // Action: 12.0 // Action: getIngredients()
13 // Action: 13.0 // Action: getIngredients()
14 // Action: 14.0 // Action: getIngredients()
15 // Action: 15.0 // Action: getIngredients()
16 // Action: 16.0 // Action: getIngredients()
17 // Action: 17.0 // Action: getIngredients()
18 // Action: 18.0 // Action: getIngredients()
19 // Action: 19.0 // Action: getIngredients()
20 // Action: 20.0 // Action: getIngredients()
21 // Action: 21.0 // Action: getIngredients()
22 // Action: 22.0 // Action: getIngredients()
23 // Action: 23.0 // Action: getIngredients()
24 // Action: 24.0 // Action: getIngredients()
25 // Action: 25.0 // Action: getIngredients()
26 // Action: 26.0 // Action: getIngredients()
27 // Action: 27.0 // Action: getIngredients()
28 // Action: 28.0 // Action: getIngredients()
29 // Action: 29.0 // Action: getIngredients()
30 // Action: 30.0 // Action: getIngredients()
31 // Action: 31.0 // Action: getIngredients()
32 // Action: 32.0 // Action: getIngredients()
33 // Action: 33.0 // Action: getIngredients()
34 // Action: 34.0 // Action: getIngredients()
35 // Action: 35.0 // Action: getIngredients()
36 // Action: 36.0 // Action: getIngredients()
37 // Action: 37.0 // Action: getIngredients()
38 // Action: 38.0 // Action: getIngredients()
39 // Action: 39.0 // Action: getIngredients()
40 // Action: 40.0 // Action: getIngredients()
41 // Action: 41.0 // Action: getIngredients()
42 // Action: 42.0 // Action: getIngredients()
43 // Action: 43.0 // Action: getIngredients()
44 // Action: 44.0 // Action: getIngredients()
45 // Action: 45.0 // Action: getIngredients()
46 // Action: 46.0 // Action: getIngredients()
47 // Action: 47.0 // Action: getIngredients()
48 // Action: 48.0 // Action: getIngredients()
49 // Action: 49.0 // Action: getIngredients()
50 // Action: 50.0 // Action: getIngredients()
51 // Action: 51.0 // Action: getIngredients()
52 // Action: 52.0 // Action: getIngredients()
53 // Action: 53.0 // Action: getIngredients()
54 // Action: 54.0 // Action: getIngredients()
55 // Action: 55.0 // Action: getIngredients()
56 // Action: 56.0 // Action: getIngredients()
57 // Action: 57.0 // Action: getIngredients()
58 // Action: 58.0 // Action: getIngredients()
59 // Action: 59.0 // Action: getIngredients()
60 // Action: 60.0 // Action: getIngredients()
61 // Action: 61.0 // Action: getIngredients()
62 // Action: 62.0 // Action: getIngredients()
63 // Action: 63.0 // Action: getIngredients()
64 // Action: 64.0 // Action: getIngredients()
65 // Action: 65.0 // Action: getIngredients()
66 // Action: 66.0 // Action: getIngredients()
67 // Action: 67.0 // Action: getIngredients()
68 // Action: 68.0 // Action: getIngredients()
69 // Action: 69.0 // Action: getIngredients()
70 // Action: 70.0 // Action: getIngredients()
71 // Action: 71.0 // Action: getIngredients()
72 // Action: 72.0 // Action: getIngredients()
73 // Action: 73.0 // Action: getIngredients()
74 // Action: 74.0 // Action: getIngredients()
75 // Action: 75.0 // Action: getIngredients()
76 // Action: 76.0 // Action: getIngredients()
77 // Action: 77.0 // Action: getIngredients()
78 // Action: 78.0 // Action: getIngredients()
79 // Action: 79.0 // Action: getIngredients()
80 // Action: 80.0 // Action: getIngredients()
81 // Action: 81.0 // Action: getIngredients()
82 // Action: 82.0 // Action: getIngredients()
83 // Action: 83.0 // Action: getIngredients()
84 // Action: 84.0 // Action: getIngredients()
85 // Action: 85.0 // Action: getIngredients()
86 // Action: 86.0 // Action: getIngredients()
87 // Action: 87.0 // Action: getIngredients()
88 // Action: 88.0 // Action: getIngredients()
89 // Action: 89.0 // Action: getIngredients()
90 // Action: 90.0 // Action: getIngredients()
91 // Action: 91.0 // Action: getIngredients()
92 // Action: 92.0 // Action: getIngredients()
93 // Action: 93.0 // Action: getIngredients()
94 // Action: 94.0 // Action: getIngredients()
95 // Action: 95.0 // Action: getIngredients()
96 // Action: 96.0 // Action: getIngredients()
97 // Action: 97.0 // Action: getIngredients()
98 // Action: 98.0 // Action: getIngredients()
99 // Action: 99.0 // Action: getIngredients()
100 // Action: 100.0 // Action: getIngredients()
    
```

What is the **output**?

Computing Choc. Chip Cookies

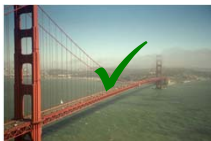


Requirements in Engineering

- Engineering is about problem solving
 - Given a set of **requirements**
 - Design a good **solution**
- If a design *does not* meet requirements
 - Not useful (in this case)
 - Wrong, broken, dangerous...
- Many designs *do* meet requirements
 - Which to choose? A “good” one, of course!
 - Optimization

Requirements: Example

- Span at least 9000'
- Support 6 lanes of traffic, 40 million car crossings per year
- Height at least 220'
- Withstand winds up to 50mph



Requirements: Example #2

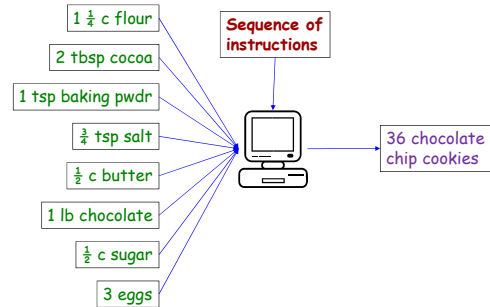
- Span at least 33'
- Support 2 lanes of pedestrian traffic
- Clearance at least 50'
- Prevent prisoners from escaping during crossing



Back to Software Engineering

- A software engineer builds *programs*
 - Instructions for **how** to turn inputs into outputs
 - Recipe engineering!
- A program must meet certain *requirements*...
 - How are requirements given for a program?
 - How are requirements given for a recipe?
 - (For software, "requirements" are usually called "specifications")

Specifying Choc. Chip Cookies



Requirements in Software

- A software engineer builds *programs*
 - Instructions for **how** to turn inputs into outputs
 - Recipe engineering!
- Programs must meet *specifications*
 - **What** transformation to do (*not* how to do it)
 - **input**: ingredients
 - **output**: final dish
- For the same requirements, many solutions
 - Good recipes are *efficient*
 - Good recipes are *fast*
 - Good recipes are *easy to understand*
 - Good recipes are *easy to change*

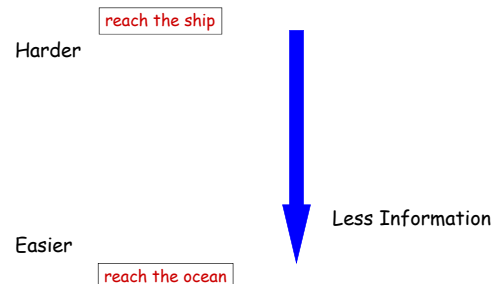
Your Turn

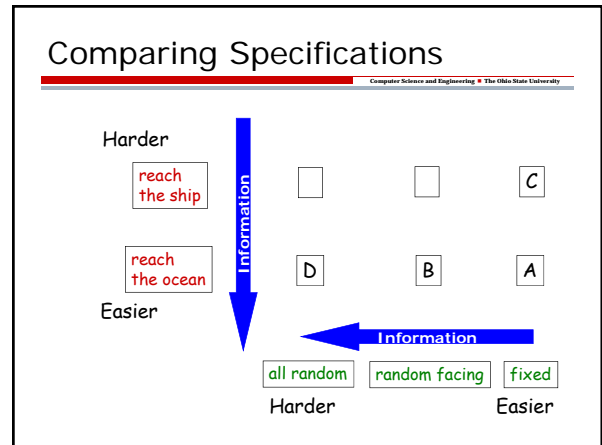
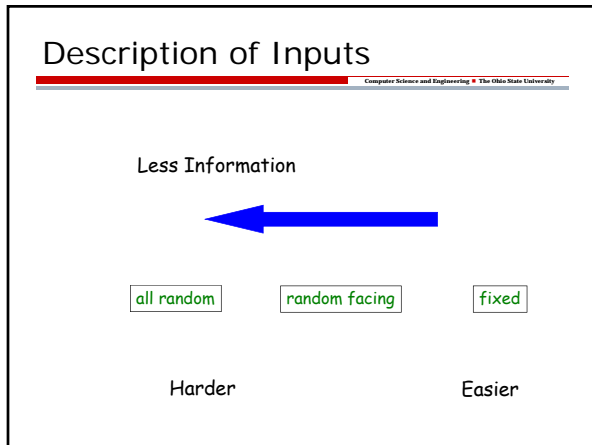
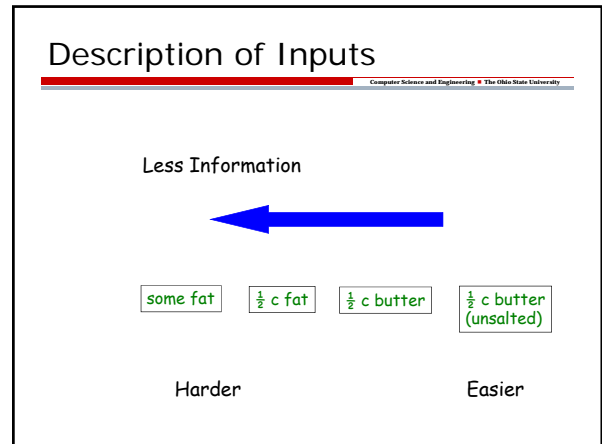
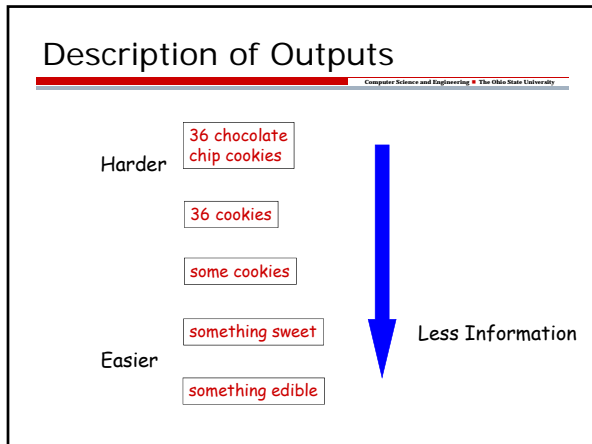
- Lab 1
 - You are given several specifications
 - Write programs that meet these specifications
- The best dishes are made from scratch...

Lab 1: Debrief

- Put the problems in increasing order of difficulty:
 - A. Fixed Start / Reach the Ocean
 - B. Random-Facing Start / Reach the Ocean
 - C. Fixed-Start / Reach the Ship
 - D. All-Random Start / Reach the Ocean
- Why is C harder than A?

Description of Outputs

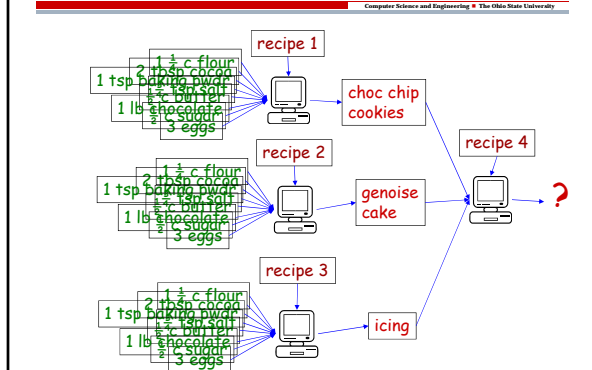




- ### Lab 1 Take-Home Messages
- Computer Science and Engineering @ The Ohio State University
- A specification that says *less* about outputs is *easier* to implement
 - But may be less useful (might not produce an appealing final dish)
 - A specification that says *less* about inputs is *harder* to implement
 - But may be more useful (more general since it can be applied in more situations)

- ### Lab 2: Composition
- Computer Science and Engineering @ The Ohio State University
- Big programs are always built out of lots of smaller ones
 - Output from one program can be used as input to another
 - Example
 - recipe for chocolate chip cookies
 - recipe for chocolate genoise cake
 - recipe for frosting

Building a Big Recipe



Take-Home Messages

- Computer Science and Engineering © The Ohio State University
- **Computer program:** a sequence of instructions
 - A recipe for a chef
 - **Specifications:** what to do (not how)
 - Given in terms of inputs and outputs
 - Less information about outputs, easier to implement
 - Less information about inputs, harder to implement
 - **Software engineering:** how to design programs
 - Recipe design: correct, easy to understand and modify
 - Usually work in teams: communication & coordination
 - **Composition:** big programs from smaller ones
 - Output of one program can be input to another

Computer Science & Engineering

Computer Science and Engineering © College of Engineering © The Ohio State University

An Introduction
(and some advanced concepts too!)

Prof. Paul Sivilotti
sivilotti.1@osu.edu