Algorithms

Slides derived from those available on the web site of the book:

<u>Computer Science: An Overview, 11th Edition, by J. Glenn Brookshear</u>

Addison-Wesley is an imprint of PEARSON

Algorithms

- The Concept of an Algorithm
- Algorithm Representation
- Algorithm Discovery
- Iterative Structures
- Recursive Structures
- Efficiency and Correctness

Informal Definition

- Our examples:
  - Converting numerical representations from one form to another
  - Detect and correct errors in data (Data Storage)
  - Control time sharing (Network)
  - Machine Cycle (CPU)
- Every activity of the human mind, is the result of algorithm execution

Definition of Algorithm

An algorithm is an **ordered** set of **unambiguous**, **executable** steps that defines a **terminating** process.

Definition of Algorithm (Cont.)

- Parallel Algorithms (e.g., Flip-Flop)
- Executable steps:
  - We cannot execute this: “Make a list of all positive integer”
- Algorithm vs Representation
- Program vs Process of *Algorithm*

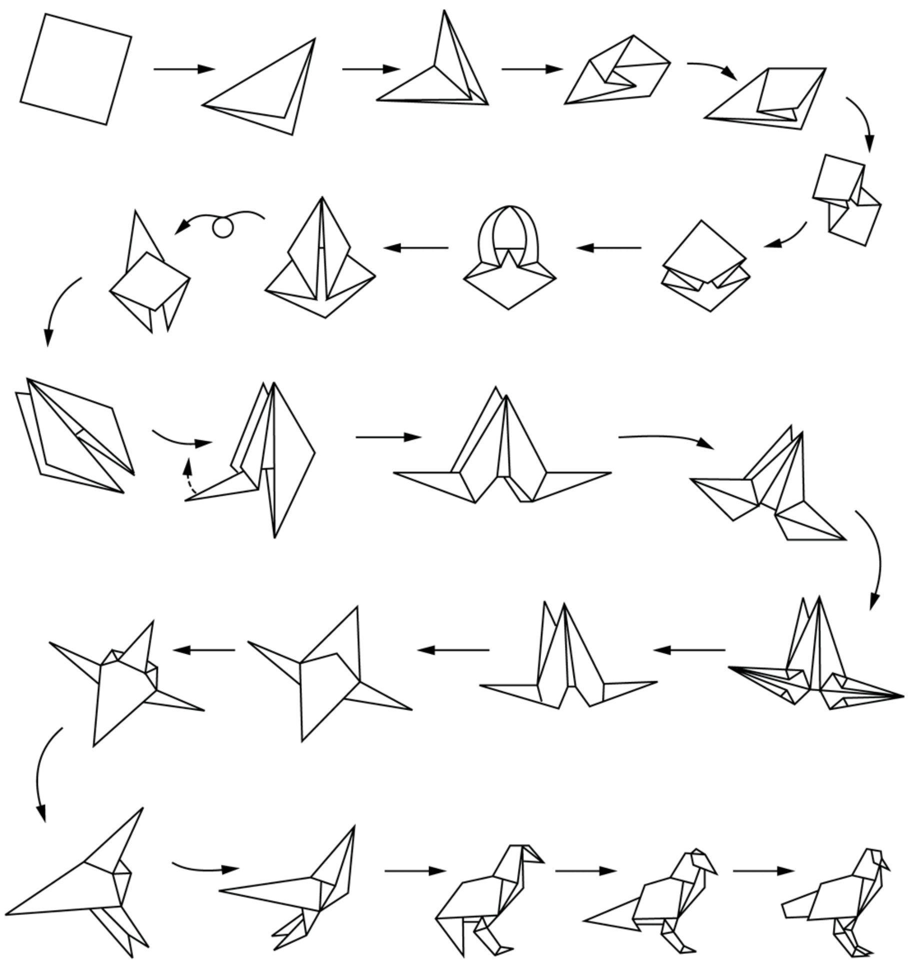
Algorithms

- The Concept of an Algorithm
- Algorithm Representation
- Algorithm Discovery
- Iterative Structures
- Recursive Structures
- Efficiency and Correctness

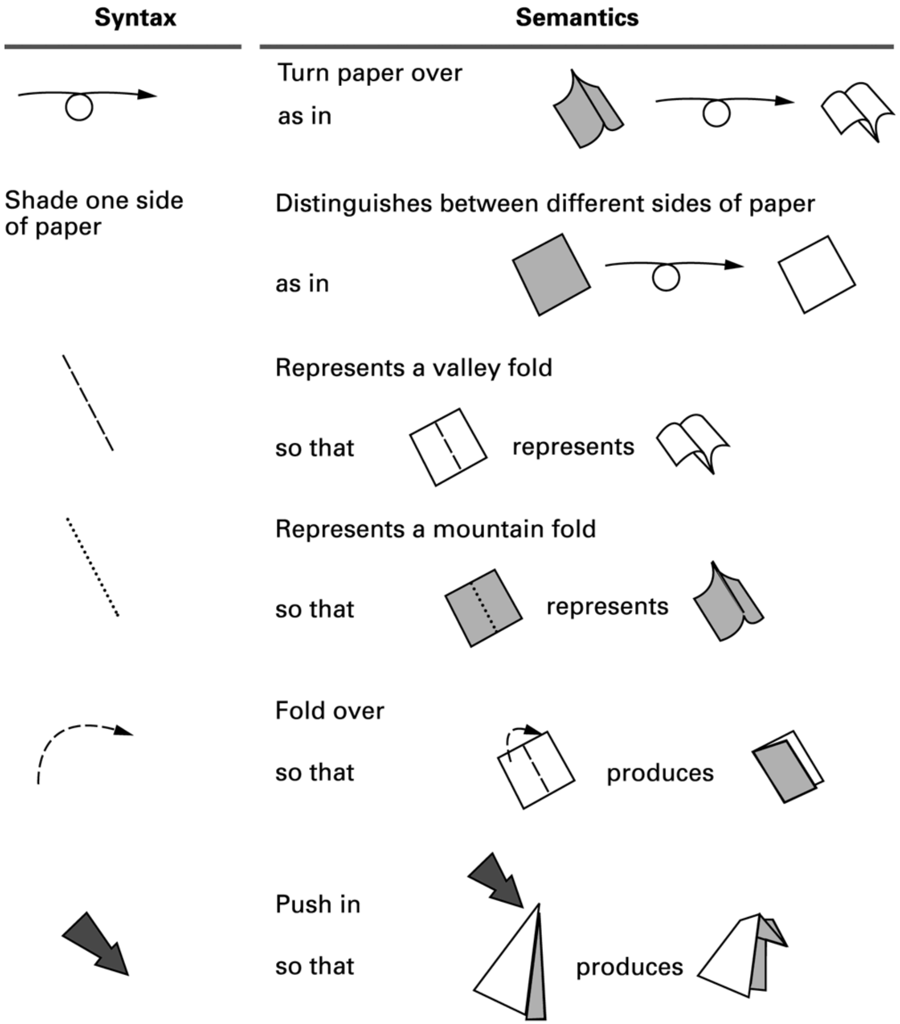
Algorithm Representation

- Requires well-defined primitives
- A collection of primitives constitutes a programming language.

Folding a bird from a square piece of paper



Origami primitives



Primitives

1. Syntax: The primitive's symbolic representation
2. Semantic: Meaning of the primitives

Pseudocode Primitives

- Assignment

    name ← *expression*

e.g., RemainingFunds ← *CheckingBalance* + *SavingBalance*

- Conditional selection

    if *condition* **then** (*action*)
    else (*action*)

Pseudocode Primitives (continued)

- Repeated execution

  while *condition* **do** *activity*

  e.g., **while** (*tickets remain to be sold)* **do** (sell a ticket)

- Procedure

  procedure *name* (*generic names*)

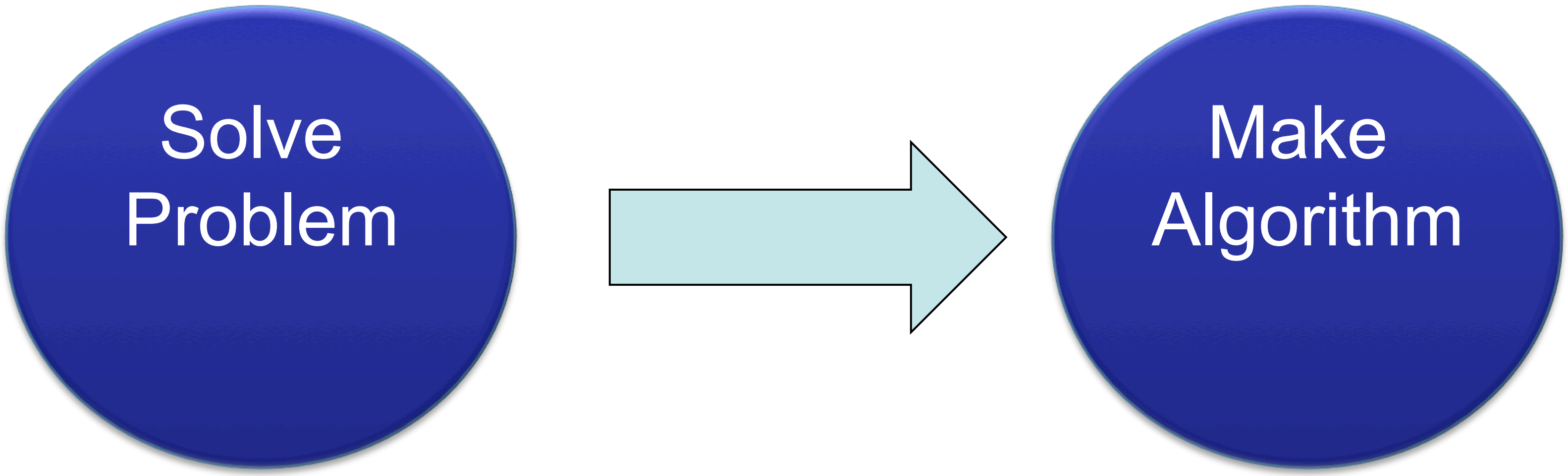
The procedure Greetings in pseudocode

```
procedure Greetings
Count ← 3;
while (Count > 0) do
    (print the message "Hello" and
        Count ← Count − 1)
```

Notes: 1. Indentation often enhances the readability
2. Naming items in programs (Pascal vs camel casing)
3. Use of (end if) and (end while)

Algorithms

- The Concept of an Algorithm
- Algorithm Representation
- Algorithm Discovery
- Iterative Structures
- Recursive Structures
- Efficiency and Correctness


Solve Problem
Make Algorithm

The Art of Solving Problem

The ability to solve problems remains more of an artistic skill to be developed than a precise science to be learned

Polya's (1945) Problem Solving Steps

1. Understand the problem.
2. Devise a plan for solving the problem.
3. Carry out the plan.
4. Evaluate the solution for accuracy and its potential as a tool for solving other problems.

Ages of Children Problem

- Person A is charged with the task of determining the ages of B's three children.
  - B tells A that the product of the children's ages is 36.
  - A replies that another clue is required.
  - B tells A the sum of the children's ages.
  - A replies that another clue is needed.
  - B tells A that the oldest child plays the piano.
  - A tells B the ages of the three children.
- How old are the three children?

Ages of Children Problem (Cont.)

a. Triples whose product is 36

(1,1,36)	(1,6,6)
(1,2,18)	(2,2,9)
(1,3,12)	(2,3,6)
(1,4,9)	(3,3,4)

b. Sums of triples from part (a)

1 + 1 + 36 = 38	1 + 6 + 6 = 13
1 + 2 + 18 = 21	2 + 2 + 9 = 13
1 + 3 + 12 = 16	2 + 3 + 6 = 11
1 + 4 + 9 = 14	3 + 3 + 4 = 10

We had not understood the problem (Phase 1)
Before start solving the problem (Phase 3)

Henri Poincare [Psychological Society, Paris]

Experience of realizing the solution to a problem he had worked on after he had set it aside and begun other projects

It is too difficult (almost impossible) to develop a systematic approach to problem solving, considering the mentioned irregularities

Let’s get a foot in the door

An example

Before A, B, C, and D ran a race they made the following predictions:

A predicted that B would win.

B predicted that D would be last.

C predicted that A would be third.

D predicted that A's prediction would be correct.

Only one of these predictions was true, and this was the prediction made by the winner. In what order did A, B, C, and D finish the race?

Getting a Foot in the Door

- Try working the problem backwards
- Solve an easier related problem
  - Relax some of the problem constraints
  - Solve pieces of the problem first (bottom up methodology)
- Stepwise refinement: Divide the problem into smaller problems (top-down methodology)

Algorithms

- The Concept of an Algorithm
- Algorithm Representation
- Algorithm Discovery
- Iterative Structures
- Recursive Structures
- Efficiency and Correctness

Iterative Structures

- Pretest loop:

```
while (condition) do
    (loop body)
```

- Posttest loop:

```
repeat (loop body)
    until(condition)
```

The sequential search algorithm in pseudocode

```
procedure Search (List, TargetValue)
if (List empty)
     then
        (Declare search a failure)
     else
        (Select the first entry in List to be TestEntry;
         while (TargetValue > TestEntry and
                     there remain entries to be considered)
              do (Select the next entry in List as TestEntry.);
         if (TargetValue = TestEntry)
              then (Declare search a success.)
              else (Declare search a failure.)
        ) end if
```

Note: List is already sorted

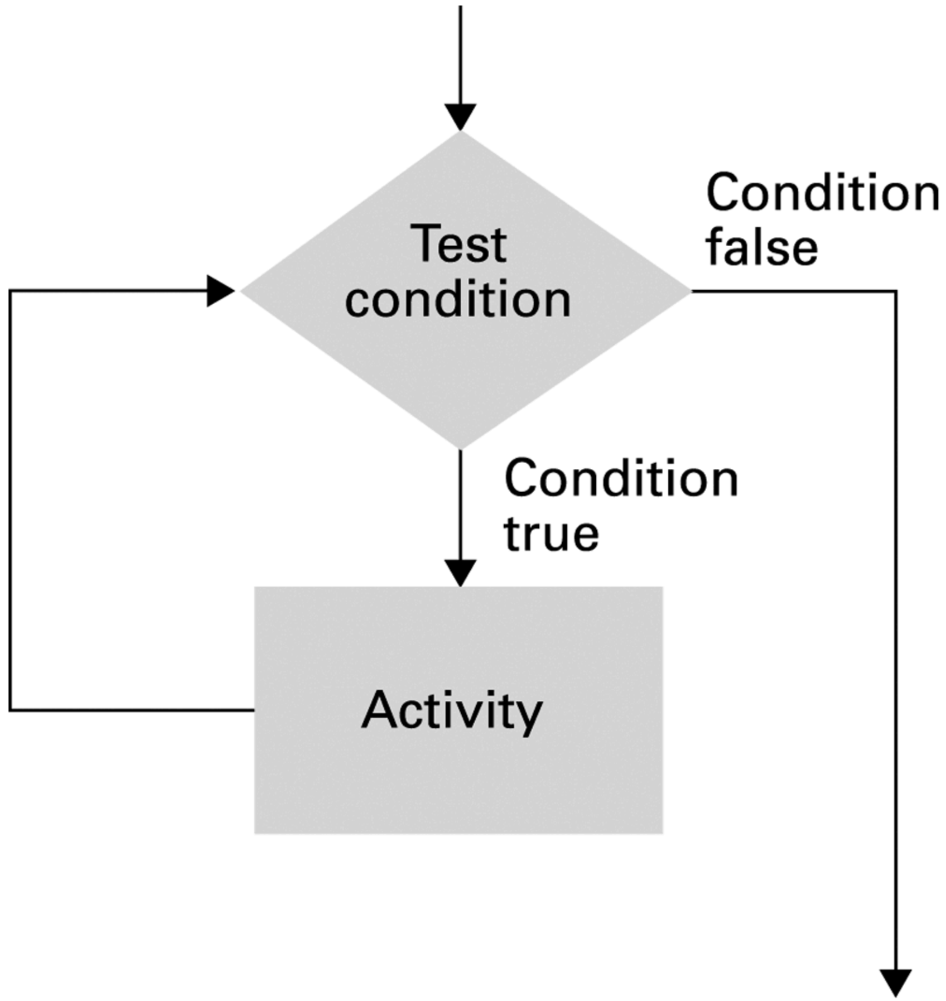
Components of repetitive control

Initialize: Establish an initial state that will be modified toward the termination condition

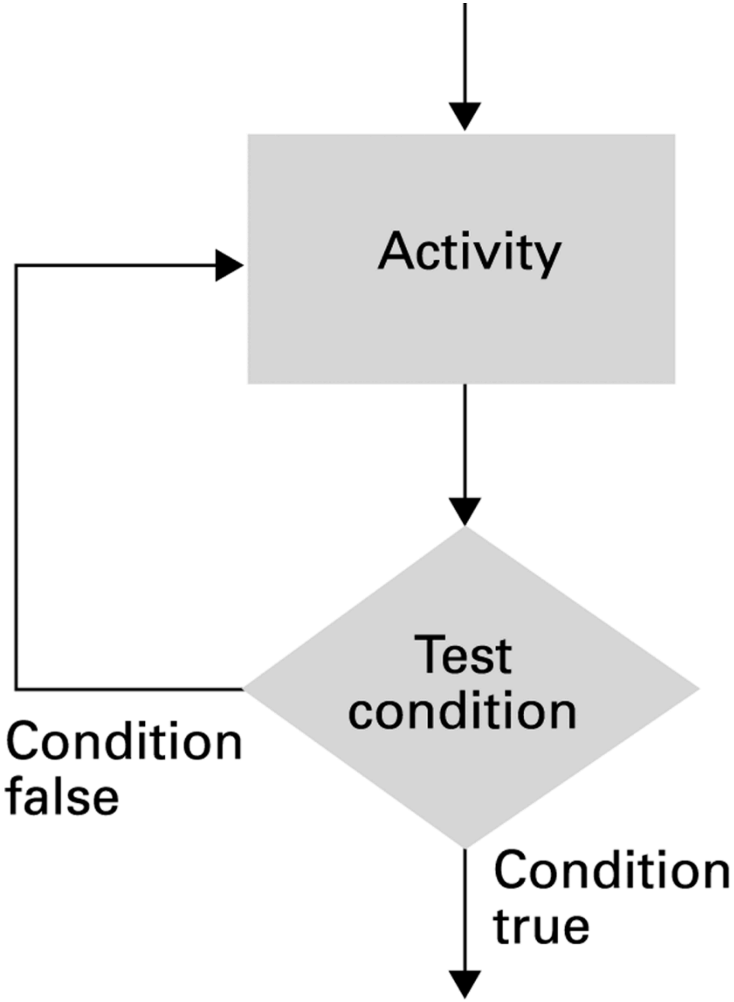
Test: Compare the current state to the termination condition and terminate the repetition if equal

Modify: Change the state in such a way that it moves toward the termination condition

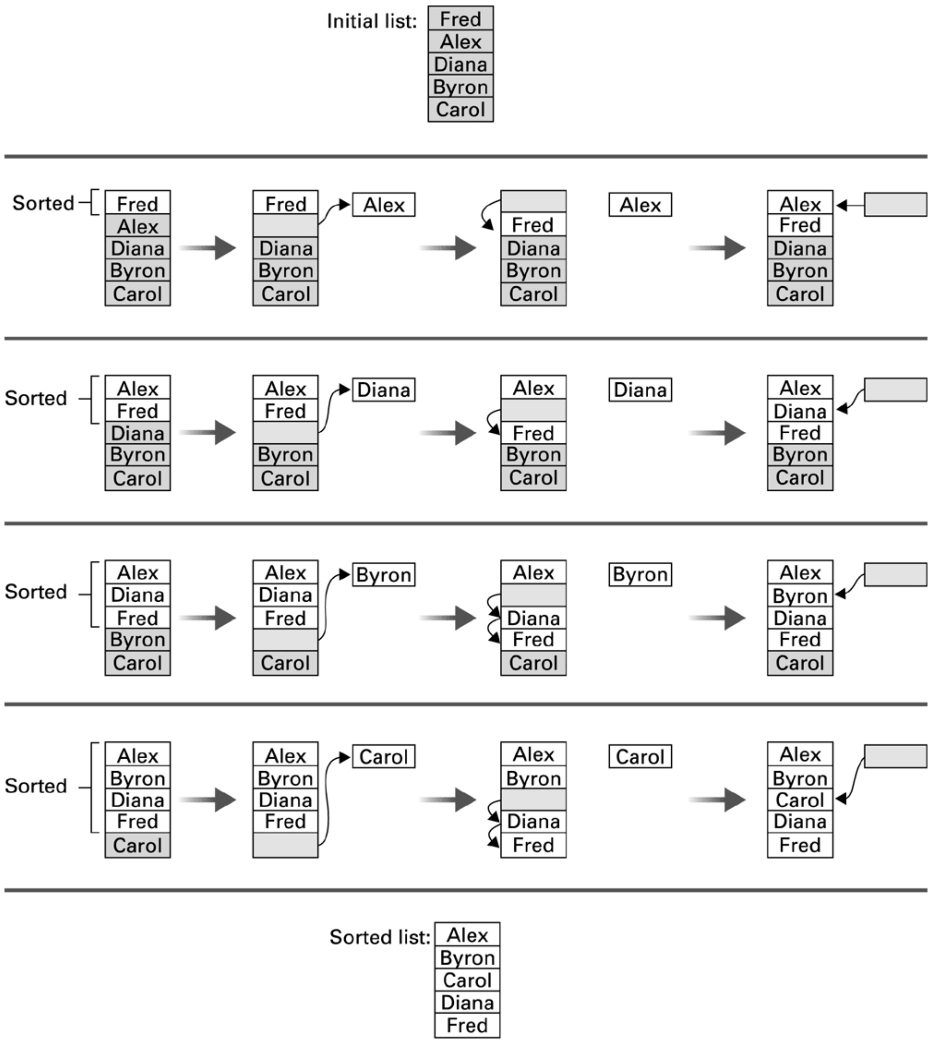
The while loop structure



The repeat loop structure



Sorting the list Fred, Alex, Diana, Byron, and Carol alphabetically



The insertion sort algorithm expressed in pseudocode

```
procedure Sort (List)
N ← 2;
while (the value of N does not exceed the length of List) do
      (Select the Nth entry in List as the pivot entry;
       Move the pivot entry to a temporary location leaving a hole in List;
        while (there is a name above the hole and that name is greater than the pivot) do
              (move the name above the hole down into the hole leaving a hole above the name)
        Move the pivot entry into the hole in List;
        N ← N + 1
      )
```

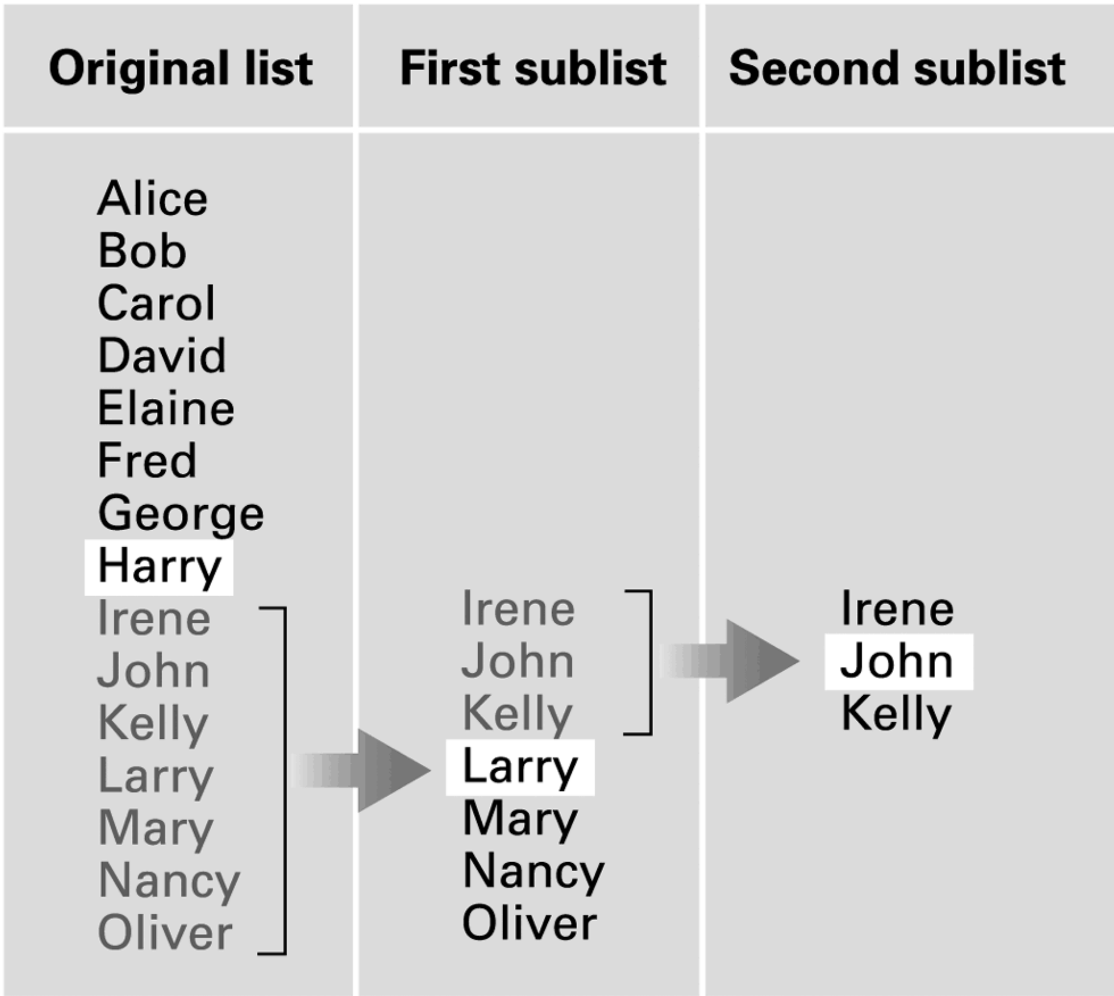
Algorithms

- The Concept of an Algorithm
- Algorithm Representation
- Algorithm Discovery
- Iterative Structures
- Recursive Structures
- Efficiency and Correctness

Recursion

- The execution of a procedure leads to another execution of the procedure.
- Multiple activations of the procedure are formed, all but one of which are waiting for other activations to complete.

Applying our strategy to search a list for the entry John



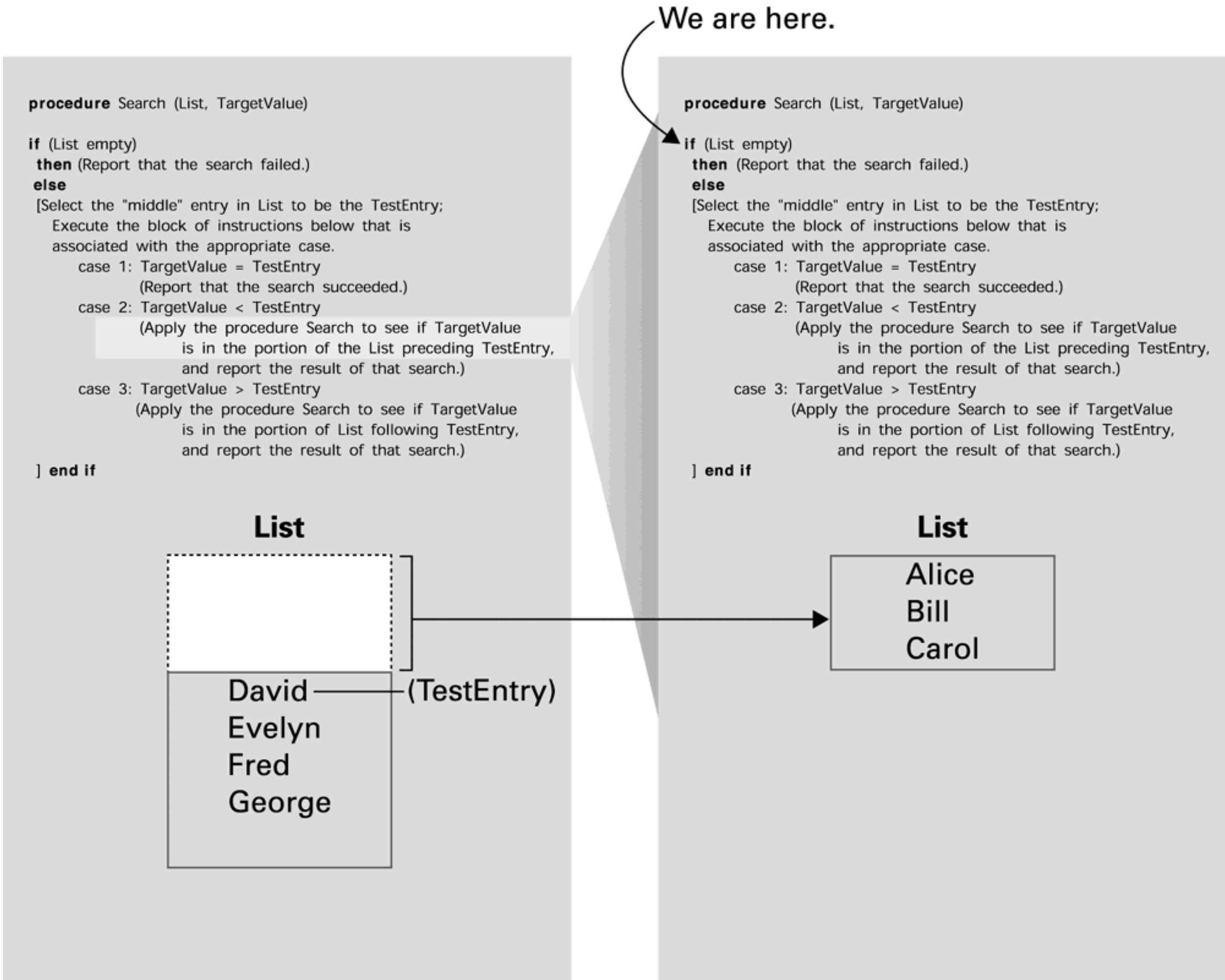
A first draft of the binary search technique

```
if (List empty)
 then
  (Report that the search failed.)
 else
   [Select the "middle" entry in the List to be the TestEntry;
    Execute the block of instructions below that is
      associated with the appropriate case.
        case 1: TargetValue = TestEntry
                (Report that the search succeeded.)
        case 2: TargetValue < TestEntry
                (Search the portion of List preceding TestEntry for
                     TargetValue, and report the result of that search.)
        case 3: TargetValue > TestEntry
                (Search the portion of List following TestEntry for
                     TargetValue, and report the result of that search.)
   ] end if
```

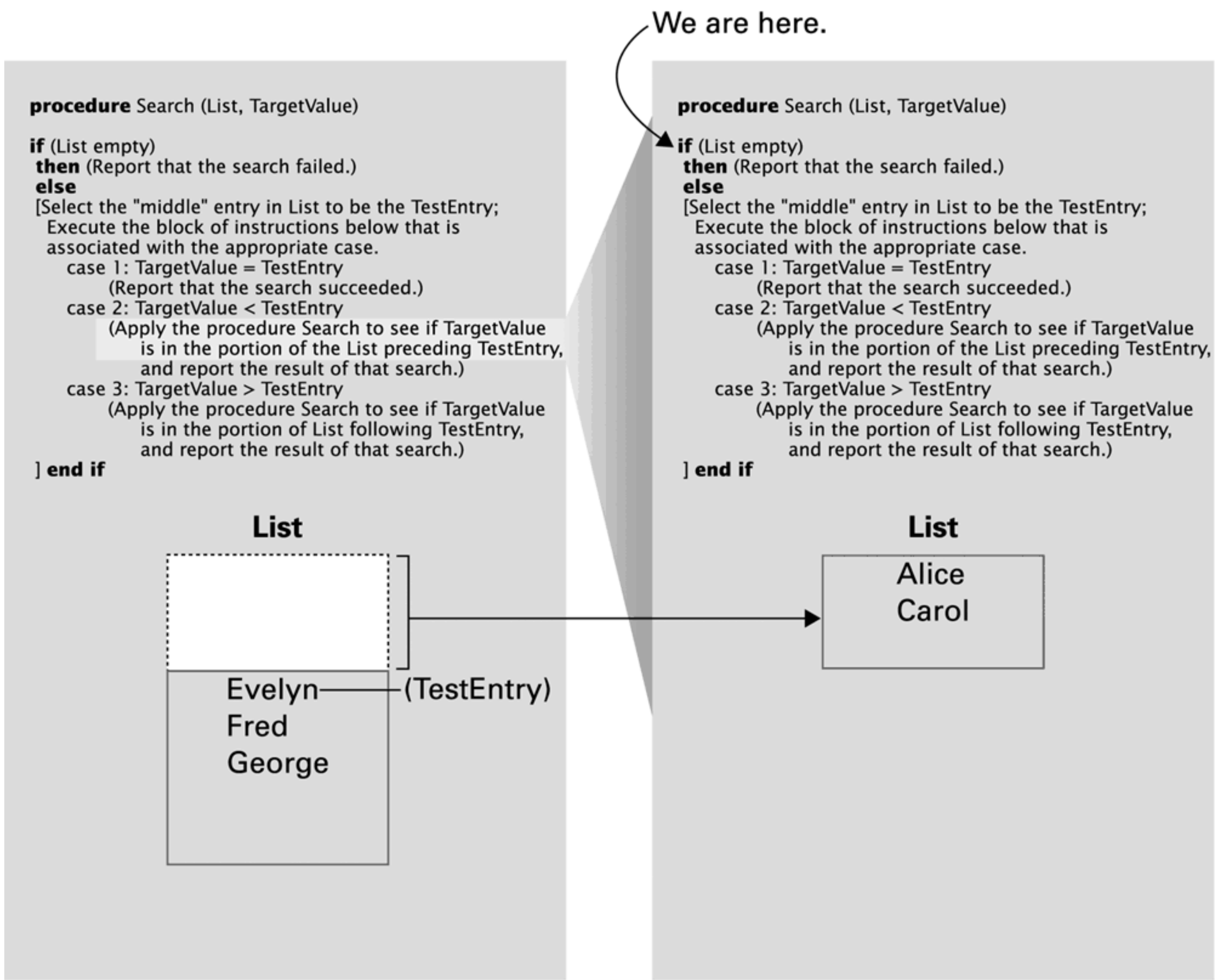
The binary search algorithm in pseudocode

```
procedure Search (List, TargetValue)
if (List empty)
  then
     (Report that the search failed.)
   else
     [Select the "middle" entry in List to be the TestEntry;
      Execute the block of instructions below that is
         associated with the appropriate case.
            case 1: TargetValue = TestEntry
                    (Report that the search succeeded.)
            case 2: TargetValue < TestEntry
                    (Apply the procedure Search to see if TargetValue
                         is in the portion of the List preceding TestEntry,
                         and report the result of that search.)
            case 3: TargetValue > TestEntry
                   (Apply the procedure Search to see if TargetValue
                        is in the portion of List following TestEntry,
                        and report the result of that search.)
     ] end if
```

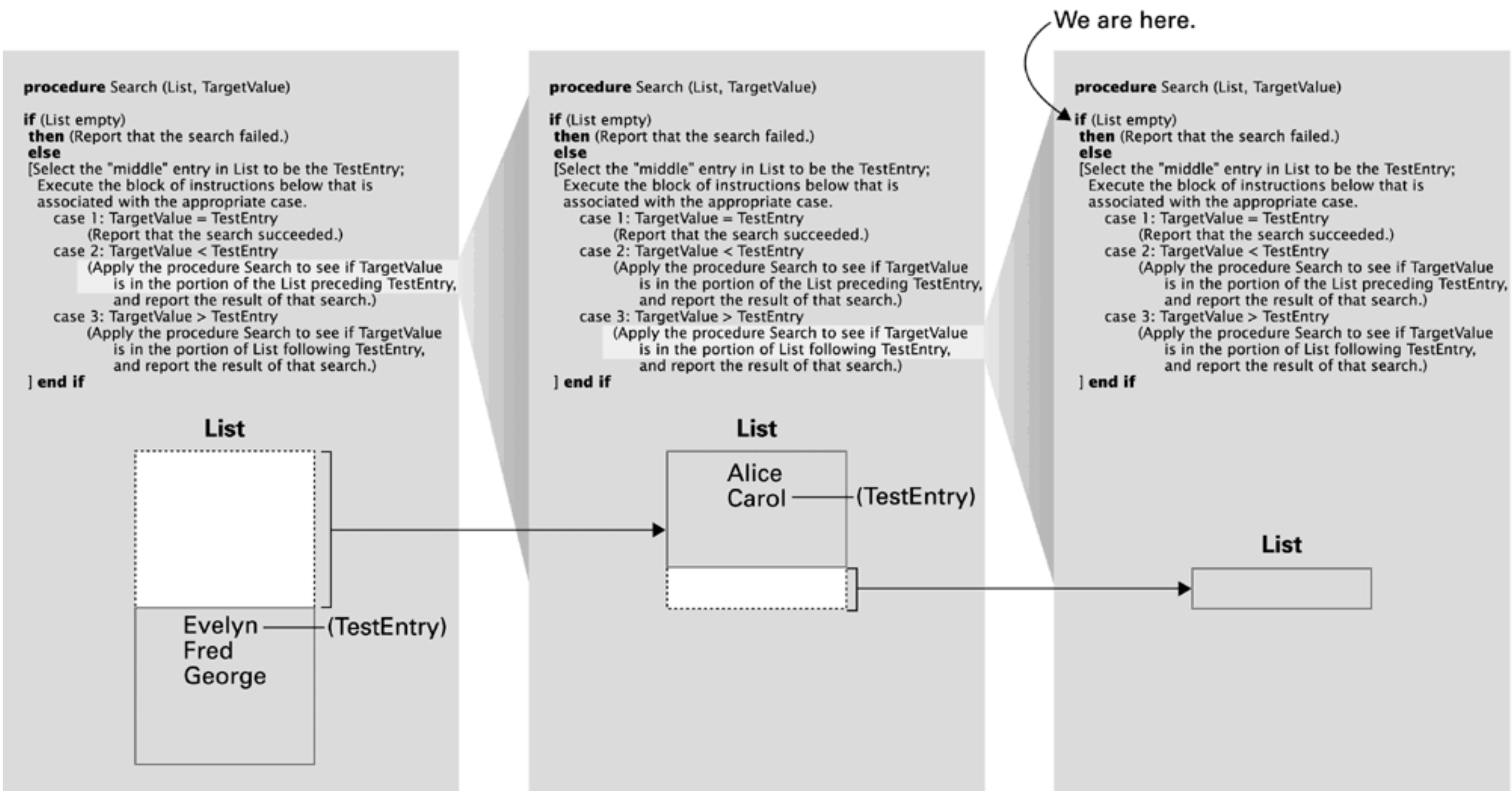
Search for “Bill”



Search for “David”



Search for “David” (Cont.)



Algorithms

- The Concept of an Algorithm
- Algorithm Representation
- Algorithm Discovery
- Iterative Structures
- Recursive Structures
- Efficiency and Correctness

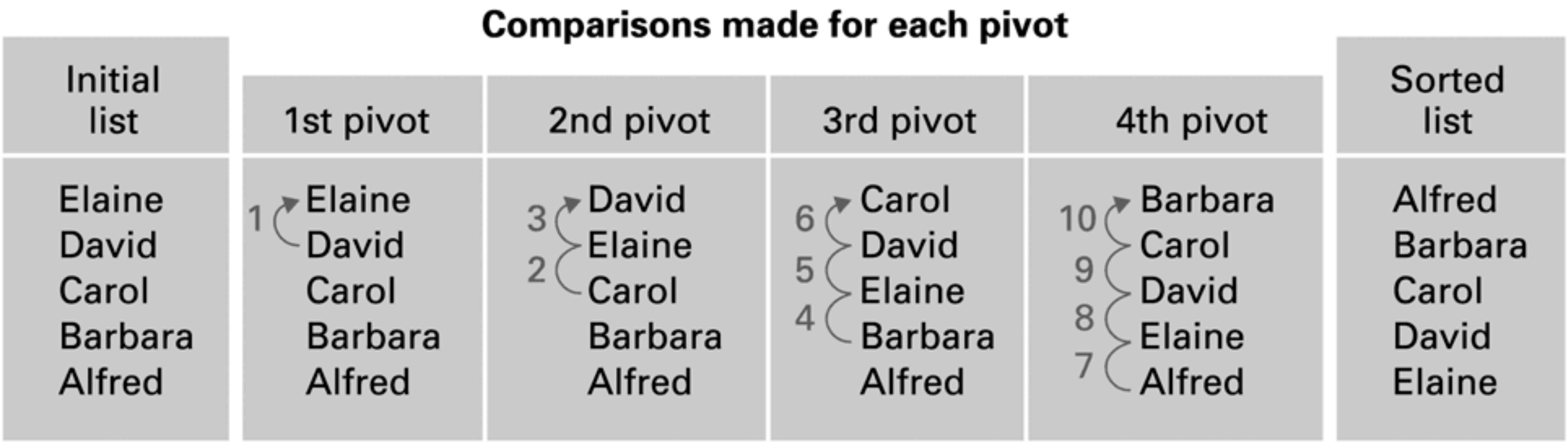
Algorithm Efficiency

- Measured as number of instructions executed
- Big theta notation: Used to represent efficiency classes
  - Example: Insertion sort is in $\Theta(n^2)$
- Best, worst, and average case analysis

Algorithm Efficiency (Example)

- Search one name in a list of 30000 students
  - Sequential Search
    - Check 15000 records per search
    - Time needed: 10ms * 15000 = 2.5 min
  - Binary Search
    - 30000 → 15000 → 7500 → 3750 → 1875 → …
    - At most 15 entries from a list of 30000
    - Time needed: 10ms * 15 = 150 ms

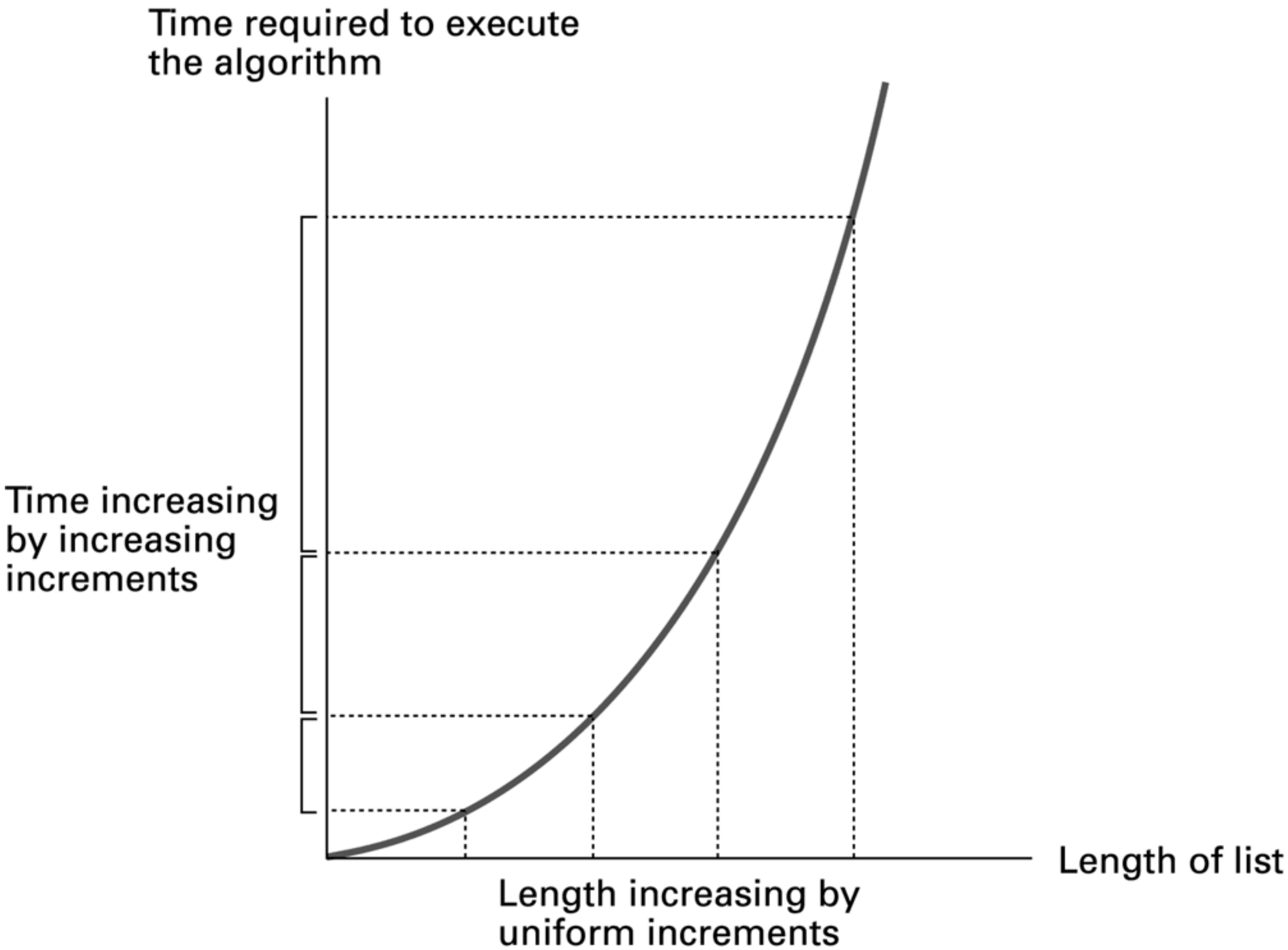
Applying the insertion sort in a worst-case situation



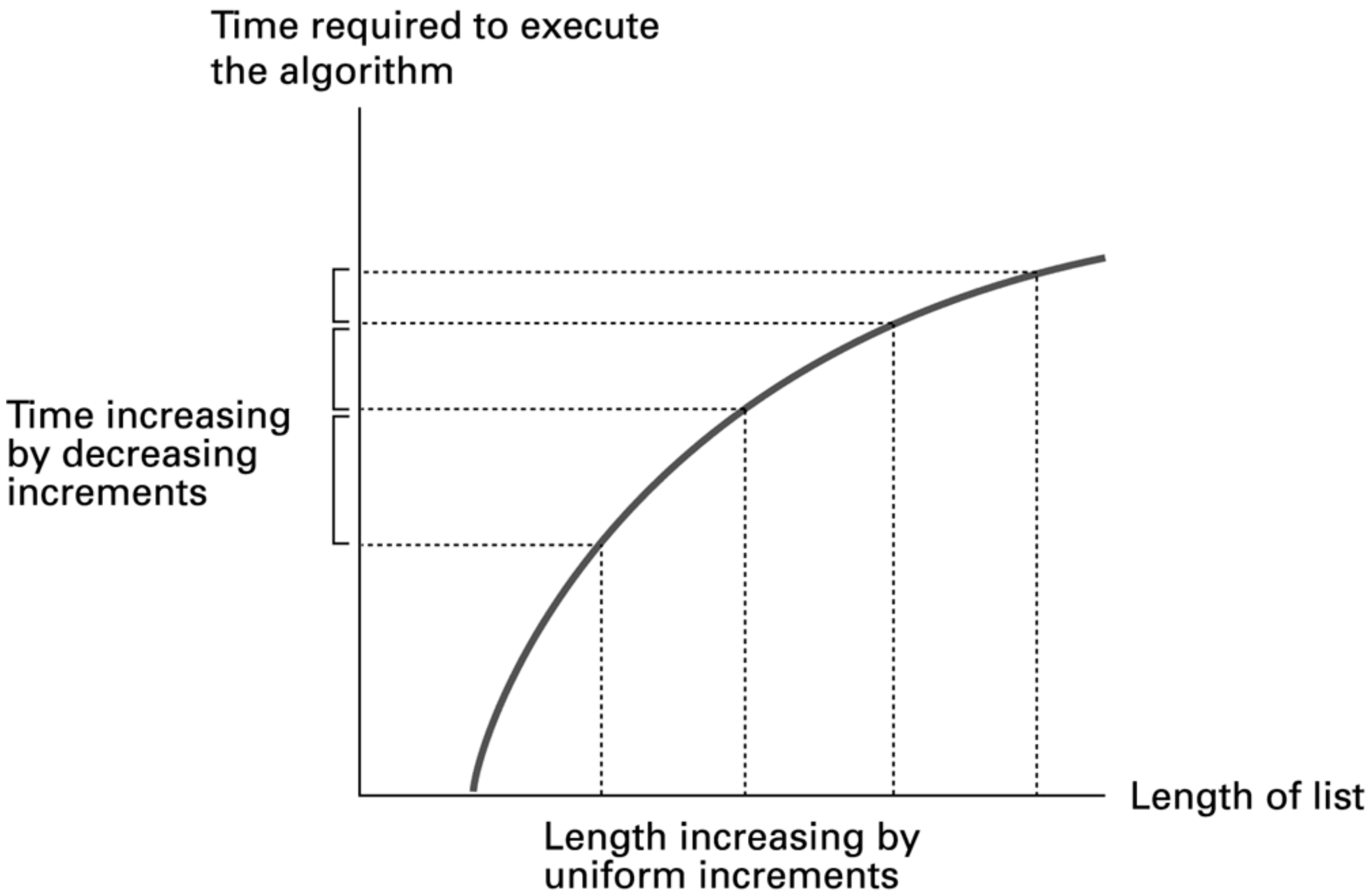
Maximum number of comparisons:

$$1+2+3 \ldots + (n-1) = \tfrac{1}{2}(n^2-n)$$

Graph of the worst-case analysis of the insertion sort algorithm



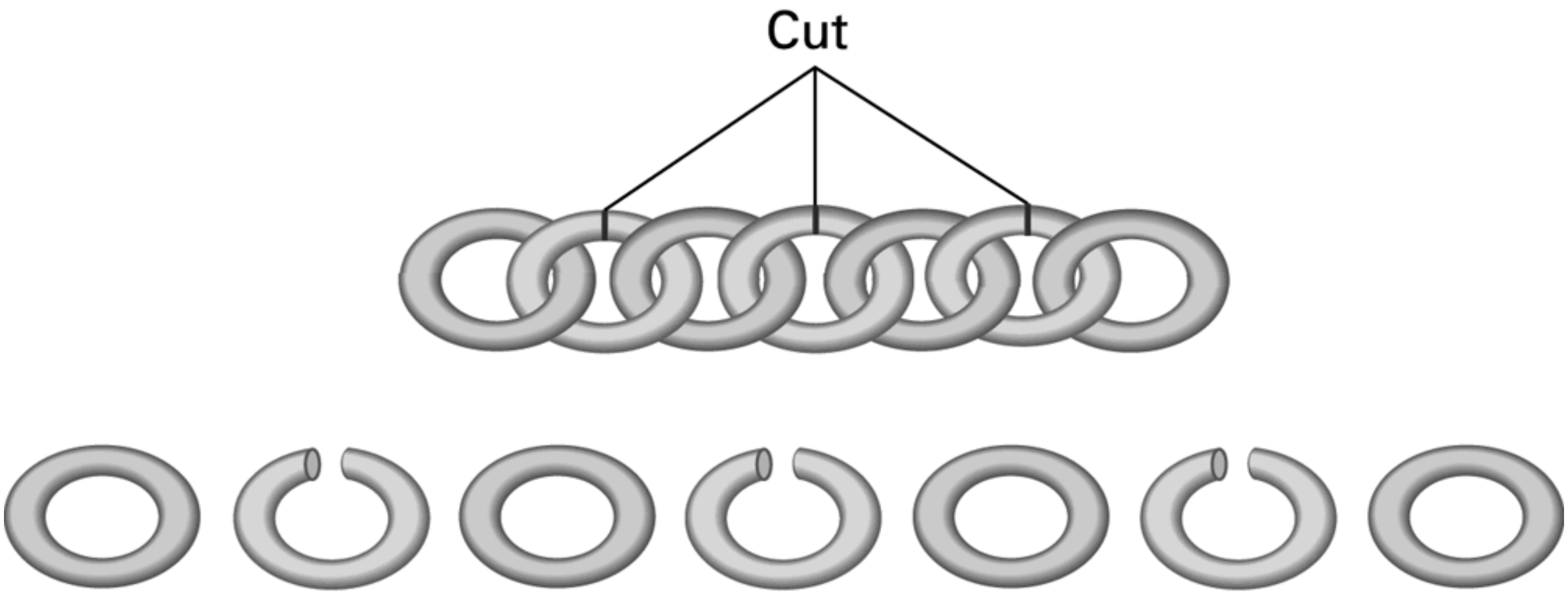
Graph of the worst-case analysis of the binary search algorithm Θ(lg(n))



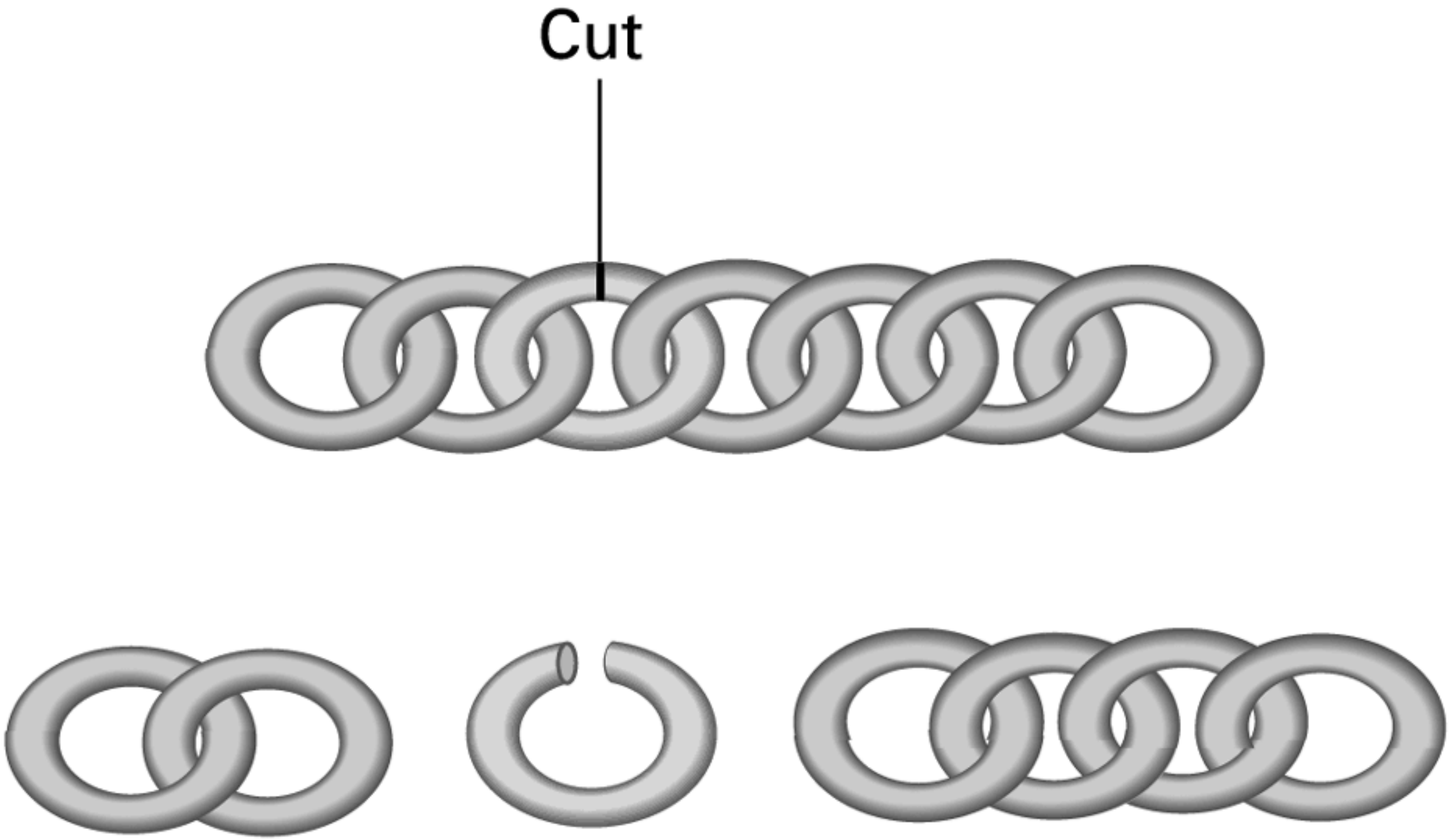
Correctness Verification Example: Chain Separating Problem

- A traveler has a gold chain of seven links.
- He must stay at an isolated hotel for seven nights.
- The rent each night consists of one link from the chain.
- What is the fewest number of links that must be cut so that the traveler can pay the hotel one link of the chain each morning without paying for lodging in advance?

Separating the chain using only three cuts



Solving the problem with only one cut



Software Verification

- Proof of correctness
  - Assertions
    - Preconditions
    - Loop invariants
- Testing