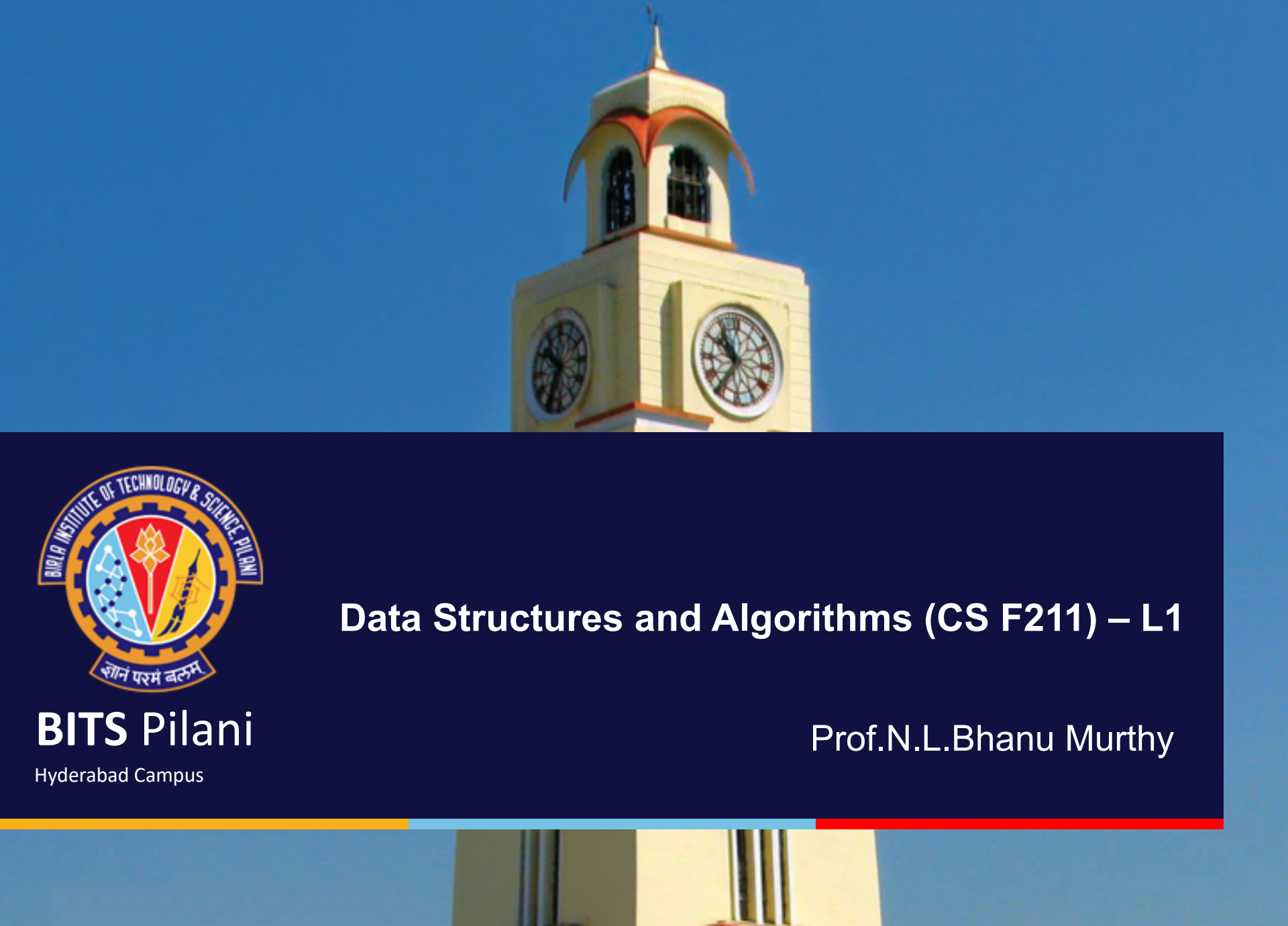


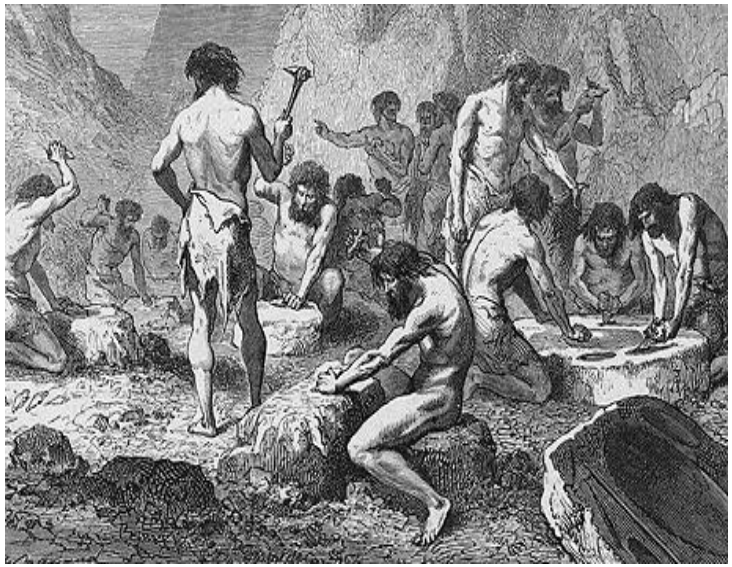
Data Structures and Algorithms (CS F211) – L1

BITS Pilani

Hyderabad Campus

Prof.N.L.Bhanu Murthy

Two discoveries that changed the world !!



Johannes Gutenberg (1398 - February 3, 1468)

Two discoveries that changed the world !!

✓ Decimal System invented in India around 600AD

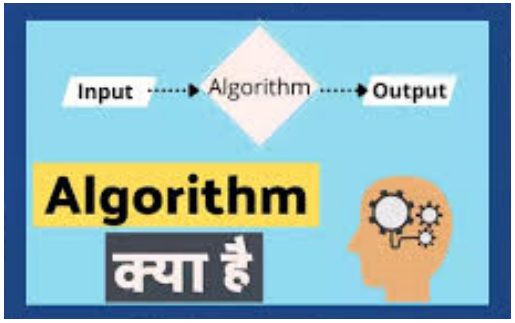
✓ Al Khwarizmi (780 AD – 850 AD) from Baghdad laid out basic methods for adding, subtracting, multiplication and dividing numbers

✓ These procedures were precise, unambiguous, mechanical, efficient, correct

✓ In short they were called **Algorithms** (a term coined to honor Al Khwarizmi - *algorithm* stem from *Algoritmi*, the Latin form of his name)

What is an algorithm?

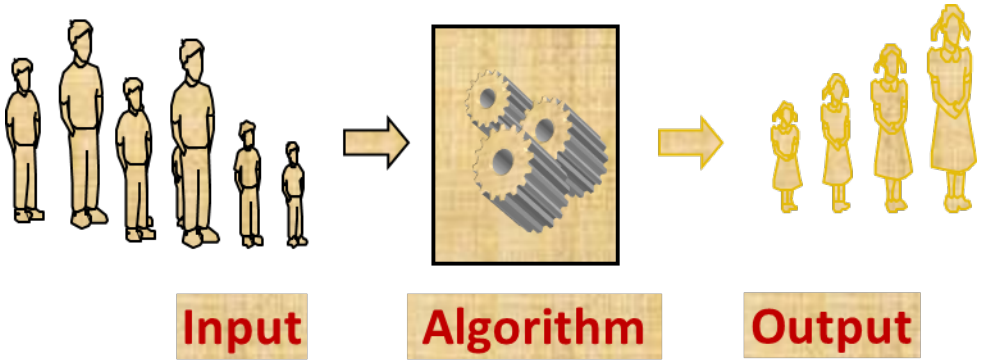
A clearly specified set of simple instructions to be followed to solve a problem



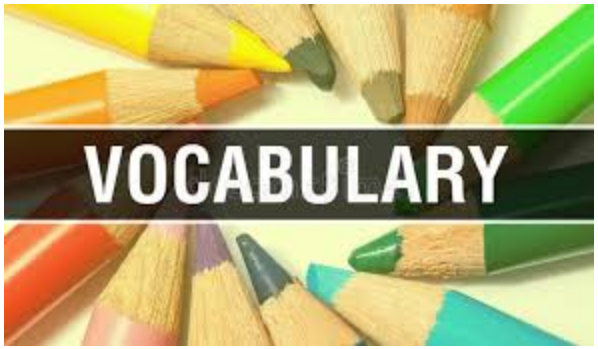
- ✓Takes a set of values, as input and
- ✓ produces a value, or set of values, as output

May be specified

- ✓In English or Hindi or Telugu
- ✓As a computer program
- ✓As a pseudo-code



Some Vocabulary with an example



Problem: Sorting of given keys

Input: A sequence of *n* keys $a_1, a_2, \ldots, a_{(n-1)}, a_n$.

Output: The permutation (reordering) of the input sequence such that $a_{\alpha(1)} \leq a_{\alpha(2)} \leq \ldots \leq a_{\alpha(n-1)} \leq a_{\alpha(n)}$.

Instance: An *instance* of sorting might be an array of names, like *{Mike, Bob, Sally, Jill ,Jan}*, or a list of numbers like *{154, 245, 568, 324, 654, 324}*

Algorithm: An *algorithm* is a procedure that takes any of the possible input instances and transforms it to the desired output.

Which algorithm is better?

Who is topper in BPHC?

Algorithm 1:

Sort *A* into decreasing order

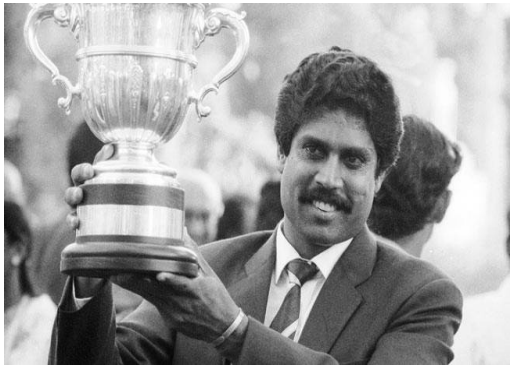
Output *A*[1].

Algorithm 2:

```
int i;
int m = A[1];
for (i = 2; i <= n; i ++)
    if (A[i] > m)
        m = A[i];
return m;
```

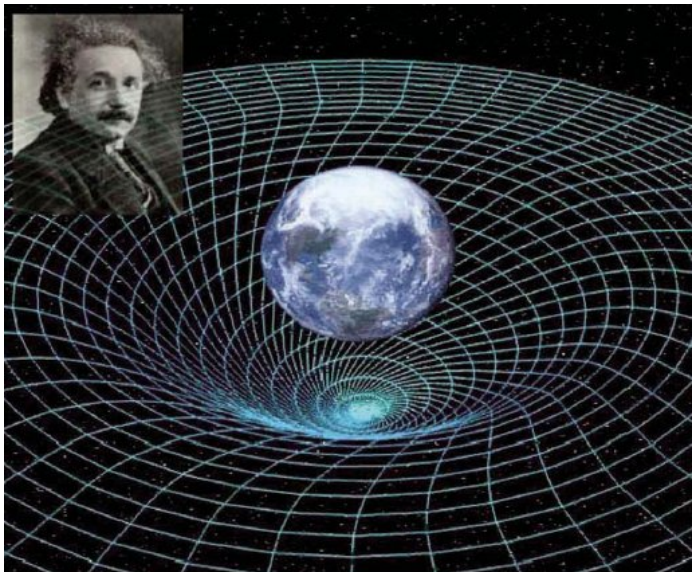
Which is better?

Who's the champion?



"Better" = more efficient

- ✓ Time
- ✓ Space



Measure efficiency (asymptotic notation)

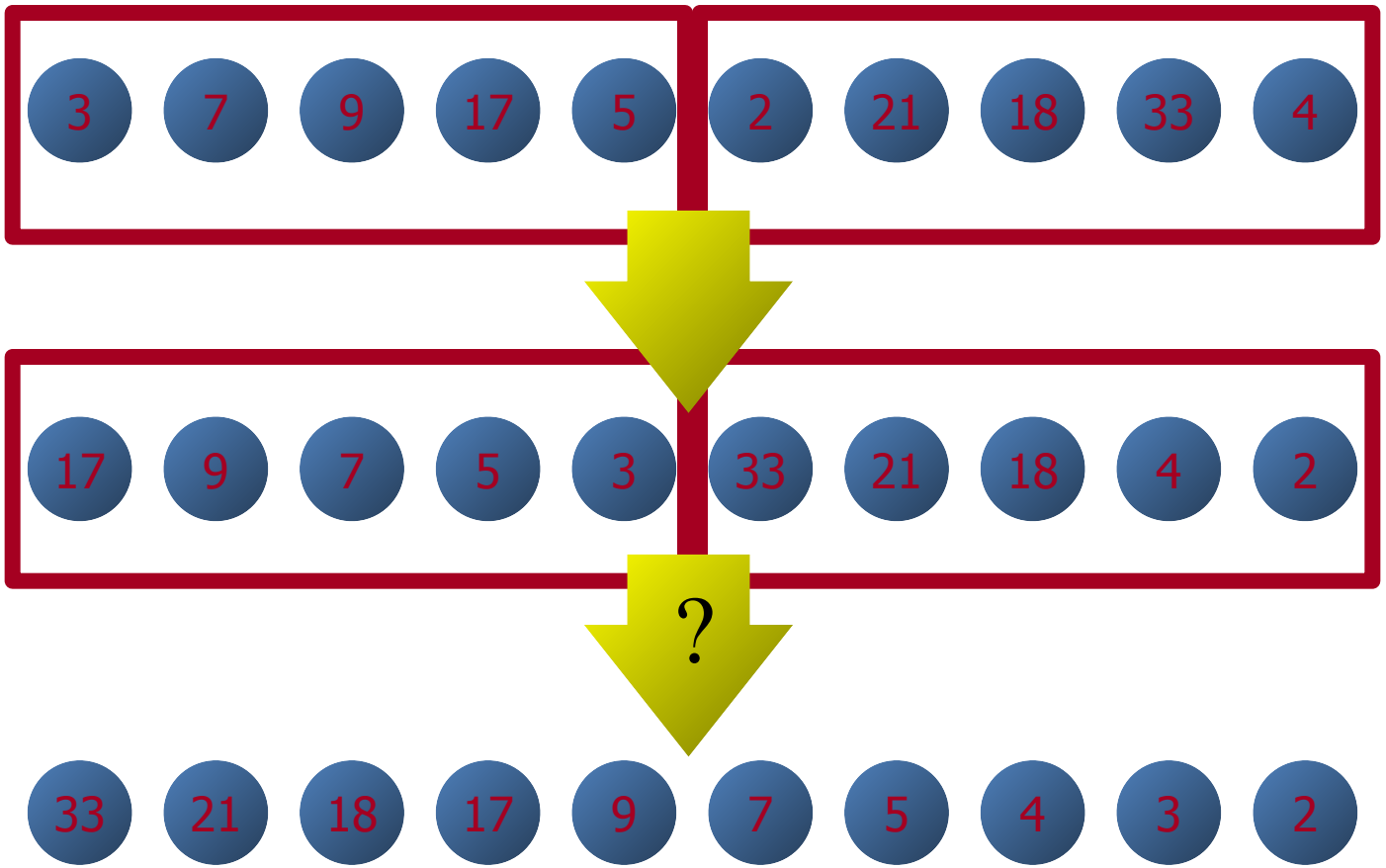
O(n)

o(n)

Ω(n)

Θ(n)

Sorting Algorithms



Comparison Based

- ✓ Bubble Sort
- ✓ Quick Sort
- ✓ Insertion Sort
- ✓ Merge Sort
- ✓ Heap Sort

Non-Comp Based

- ✓ Radix Sort
- ✓ Bucket Sort

Lower bound on comparison based algorithms

Data Structures

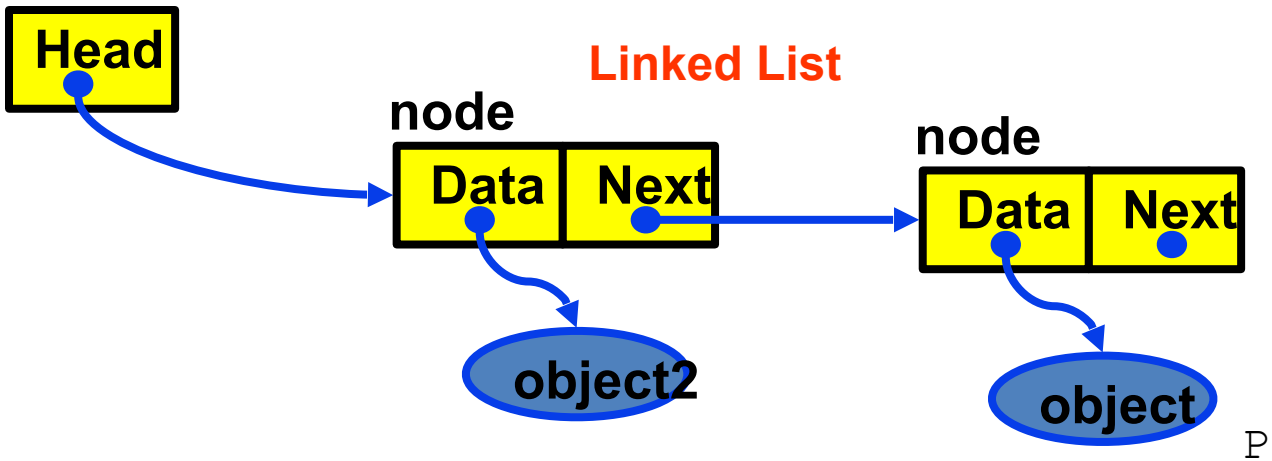
Name of array (Note that all elements of this array have the same name, **c**)

Array

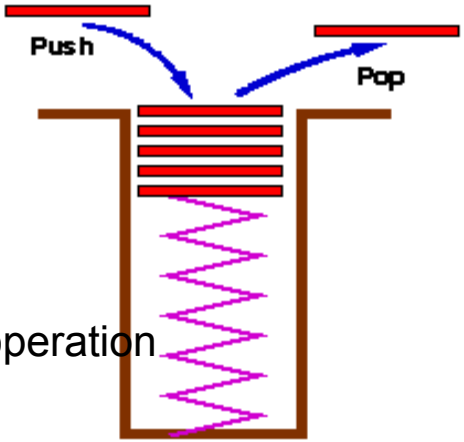
Linearly Ordered Set

c[0]	-45
c[1]	6
c[2]	0
c[3]	72
c[4]	1543
c[5]	-89
c[6]	0
c[7]	62
c[8]	-3
c[9]	1
c[10]	6453
c[11]	78

Position number of the element within array **c**

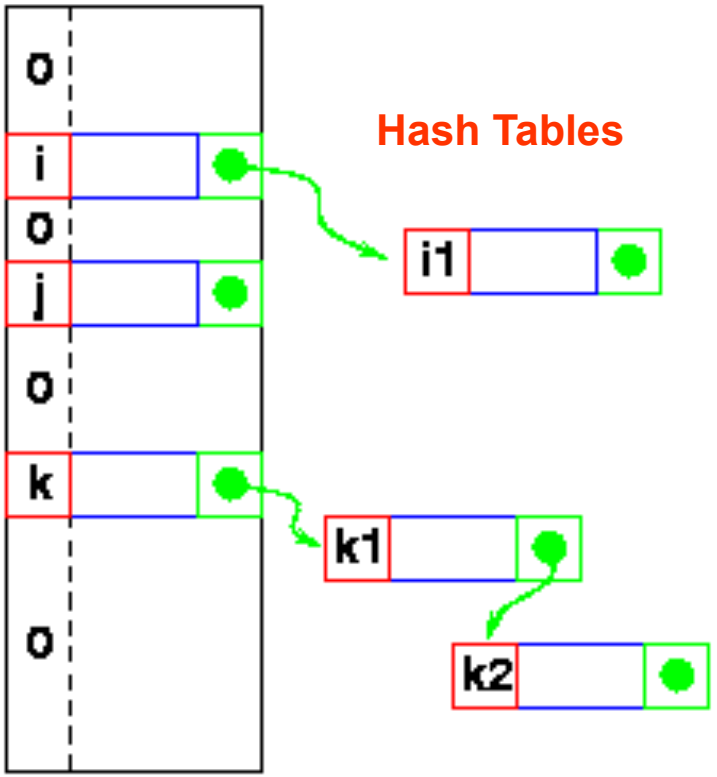


Data Structures



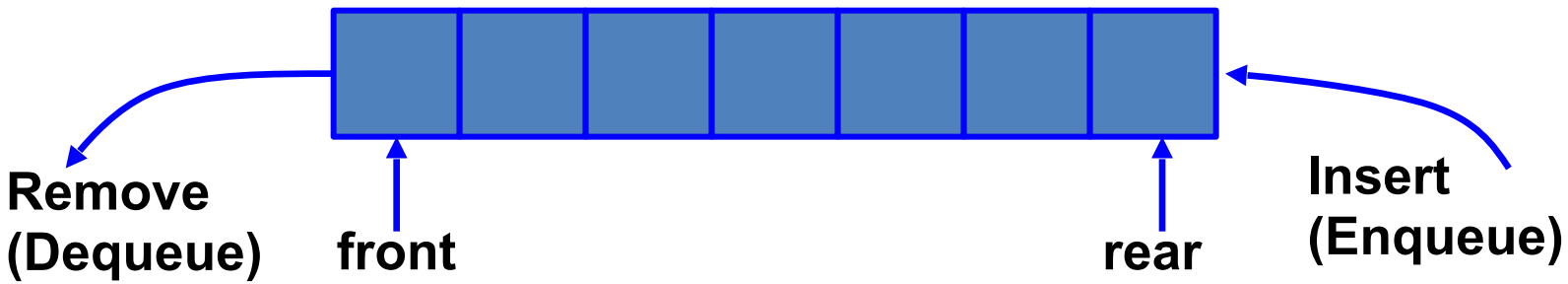
Stack

Set with delete operation specified (LIFO)



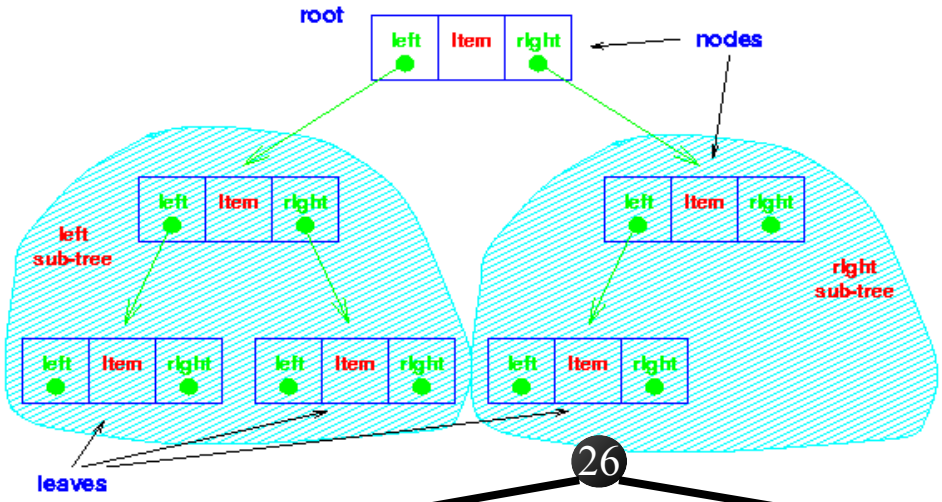
Hash Tables

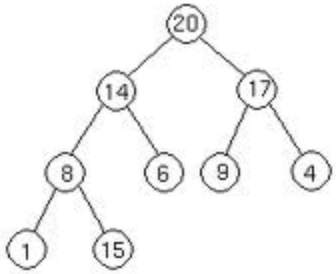
Queue



Data Structures

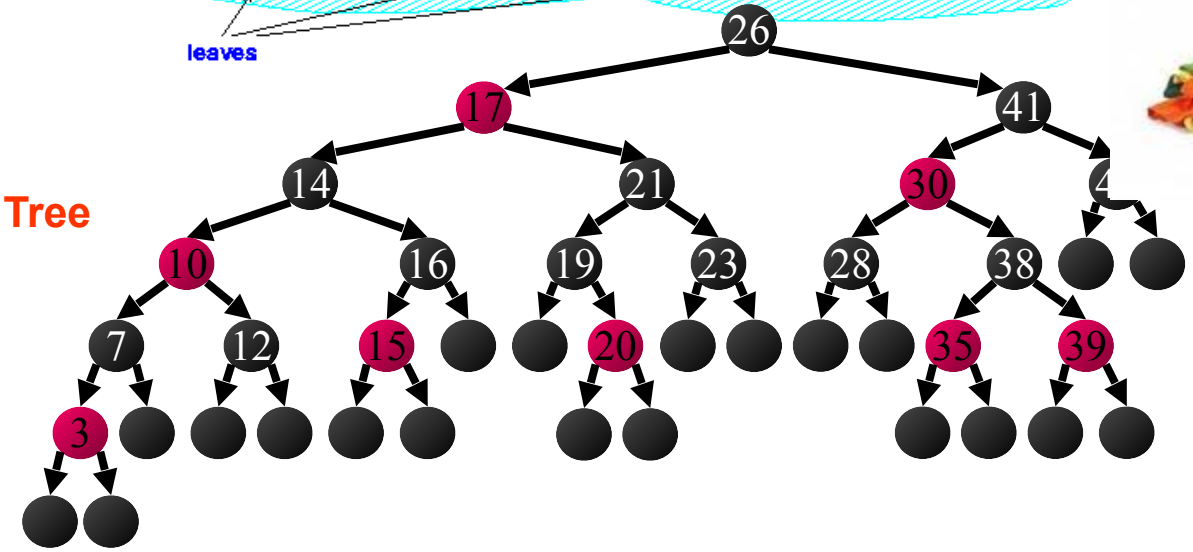
Binary Tree





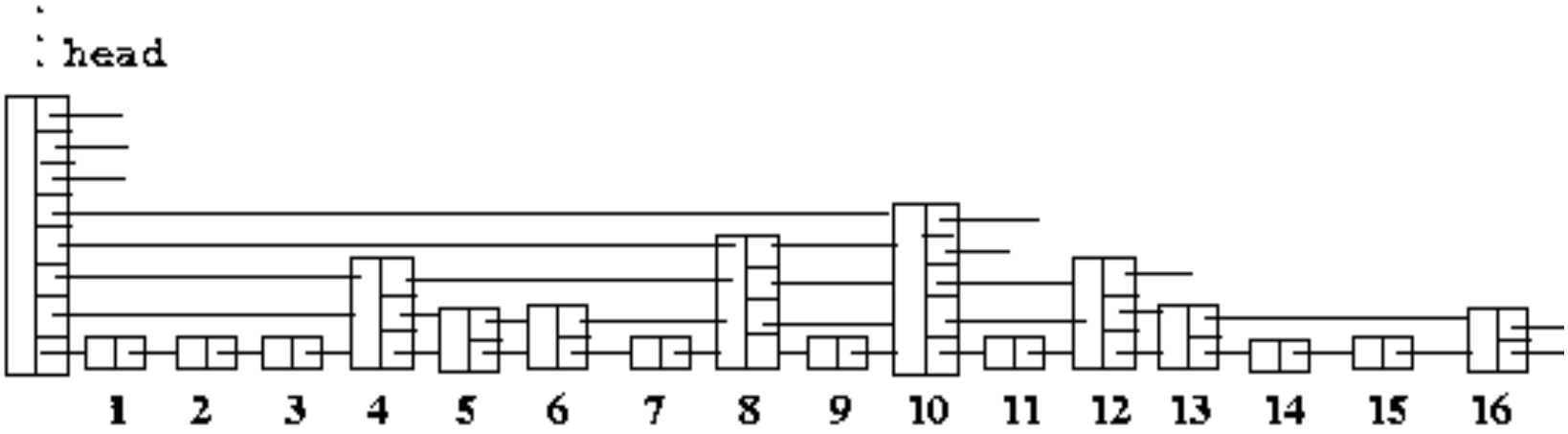
Heap

Red Black Tree



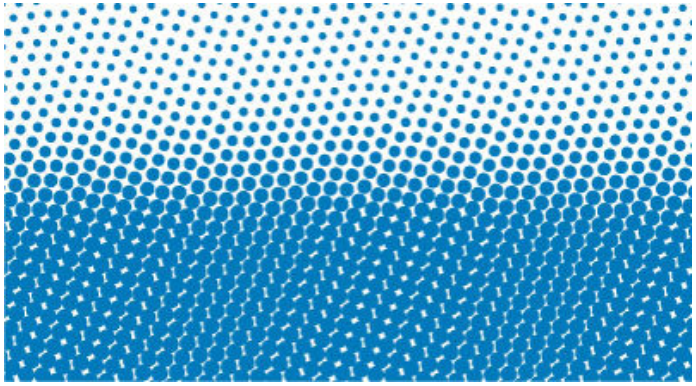
Data Structures

Skip List



Algorithm Techniques – Divide and Conquer

Nearest Points



Matrix Multiplication

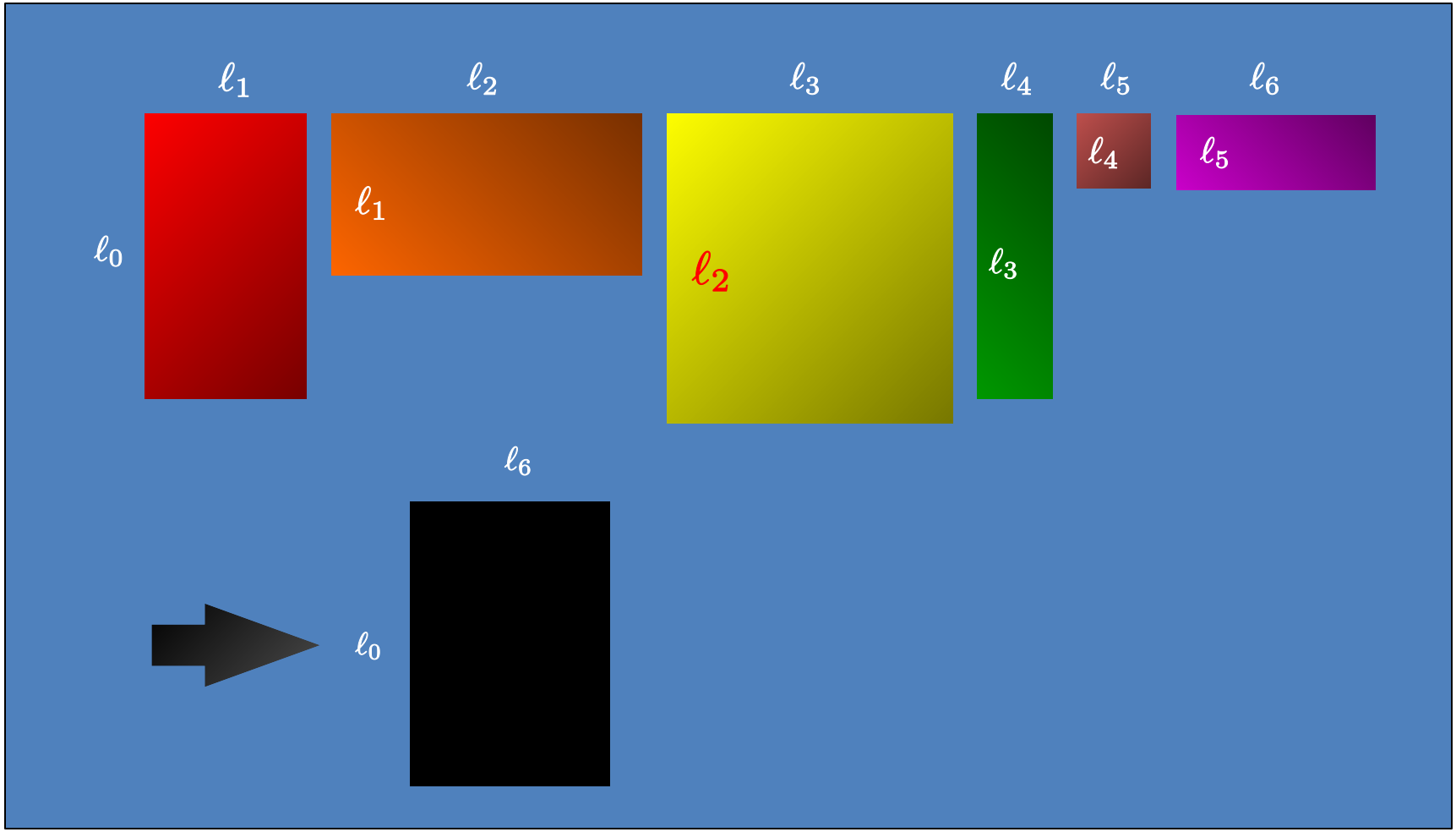
Algorithm Techniques – Dynamic Programming

Efficient algorithm to compute $F(n)$

Leonardo Fibonacci
1170-1250

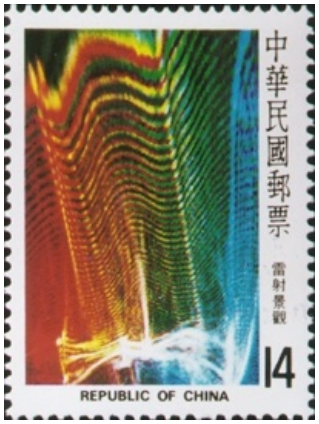
Algorithm Techniques – Dynamic Programming

Matrix Chain Product



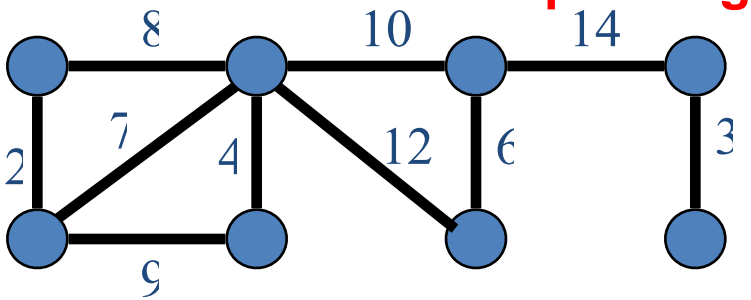
Algorithm Techniques – Dynamic Programming

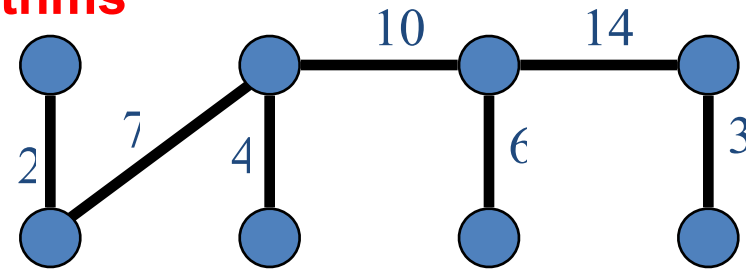
If the postage is *n*, what is the minimum number of stamps to cover the postage?



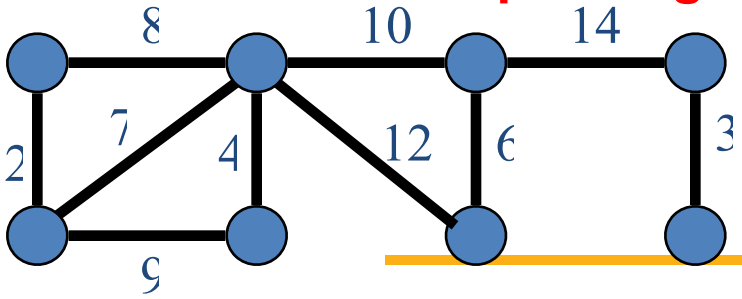
Algorithm Techniques – Greedy Approach

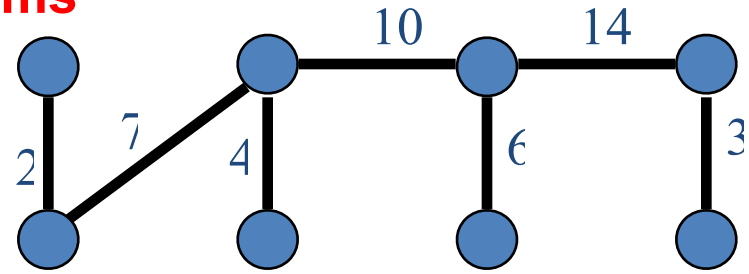
Krusal's Minimum Spanning Tree Algorithms





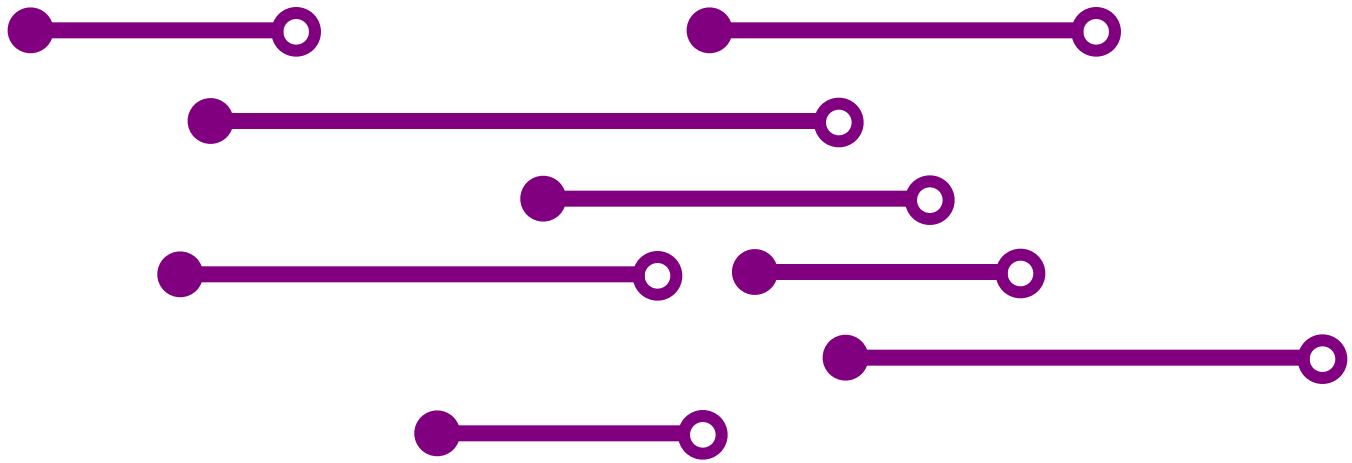
Prim's Minimum Spanning Tree Algorithms





Algorithm Techniques – Greedy Approach

Task Selection Problem: Selecting as many disjoint tasks as possible.



P, NP, NP-Complete, NP Hard

Complexity Class P

Set of all decision problems (or languages) that can be solved in polynomial time

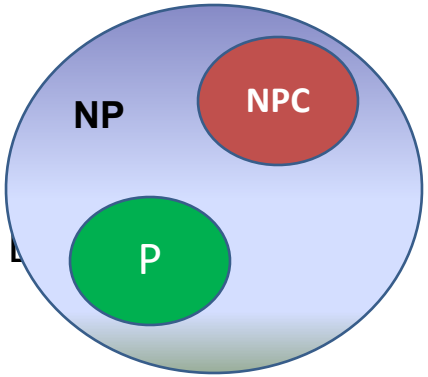
Complexity Class NP

Set of all decision problems (or languages) that can be verified by a polynomial-time algorithm

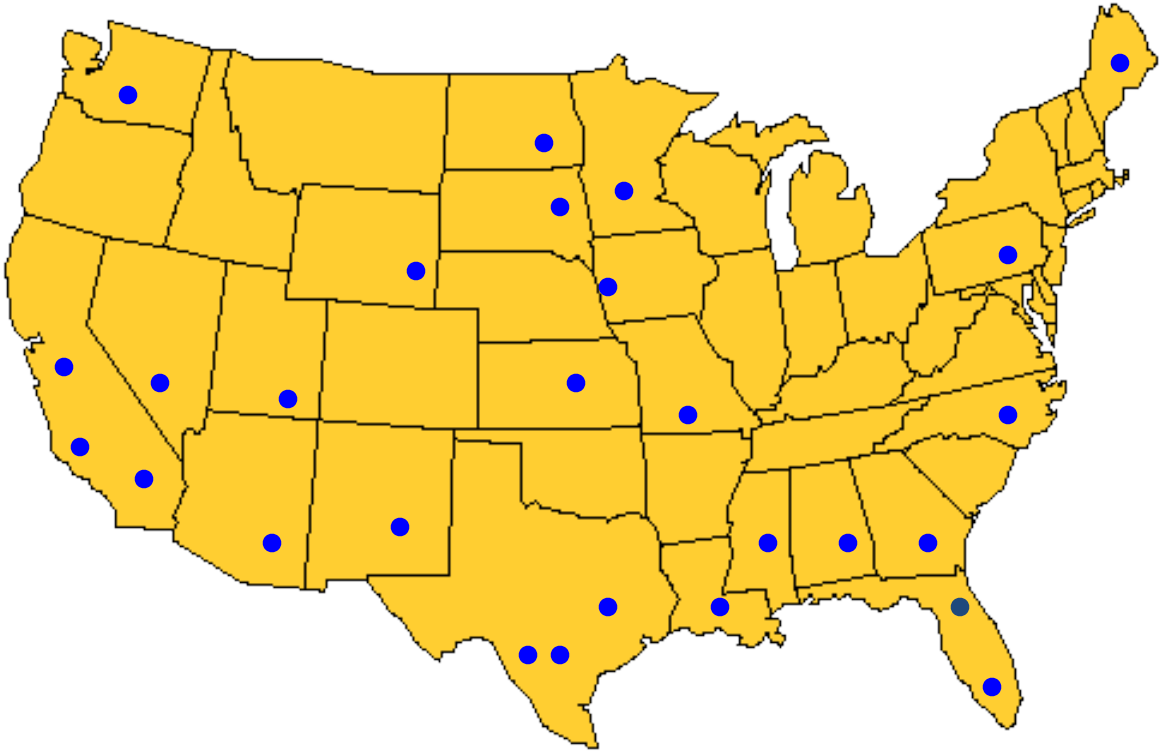
NP-completeness

A language L is NP-complete if

- ✓ L is in NP and
- ✓ All other languages in NP are polynomially reducible to L



P, NP, NP-Complete, NP Hard

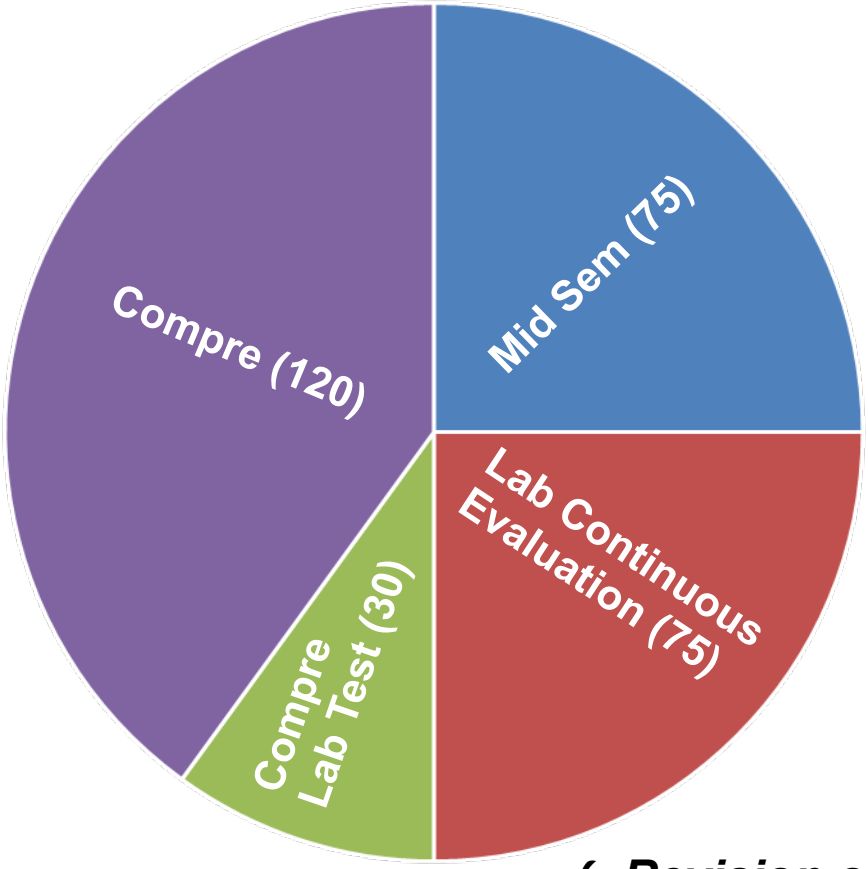


- Home city
- Visit city

Millennium problems (US $1,000,000 per problem)

- ✓ Birch and Swinnerton-Dyer Conjecture
- ✓ Hodge Conjecture
- ✓ Navier-Stokes Equations
- ✓ **Is P = NP?**
- ✓ Poincaré Conjecture
- ✓ Riemann Hypothesis
- ✓ Yang-Mills Theory

Evaluation (300 Marks)



✓ ***Revision of advanced C***

Teaching & Evaluation

CS F 211 – L P U – 3 1 4

Team

Prof. N.L.BHANU MURTHY (Lecture)
Dr. Barsh Mitra (2 Labs)
Dr. Sameera 2 Labs)
Dr. Paresh Saxena (2 Tutorial)
Dr. Raghunath Reddy

Make-up Policy

No makeup for quizzes, lab assignments and lab test.

Make-up for other tests will be granted on prior permission and on justifiable grounds only.

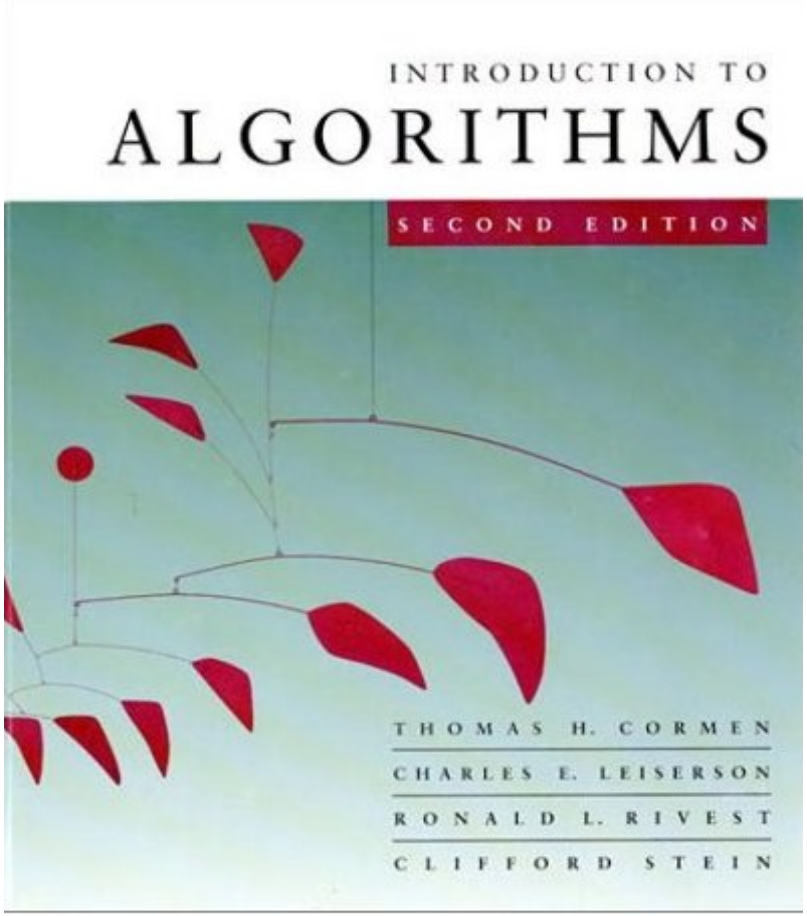
Course Notices

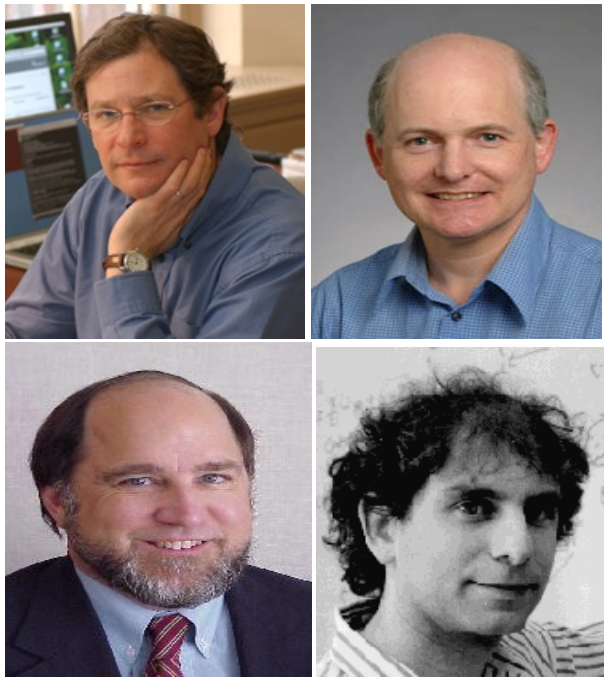
All notices pertaining to this course will be displayed on the LTC Notice Board as well as the CS & IS Notice Board.

Chamber Consultation

Tuesday 1600Hrs – 1700Hrs

Text Book



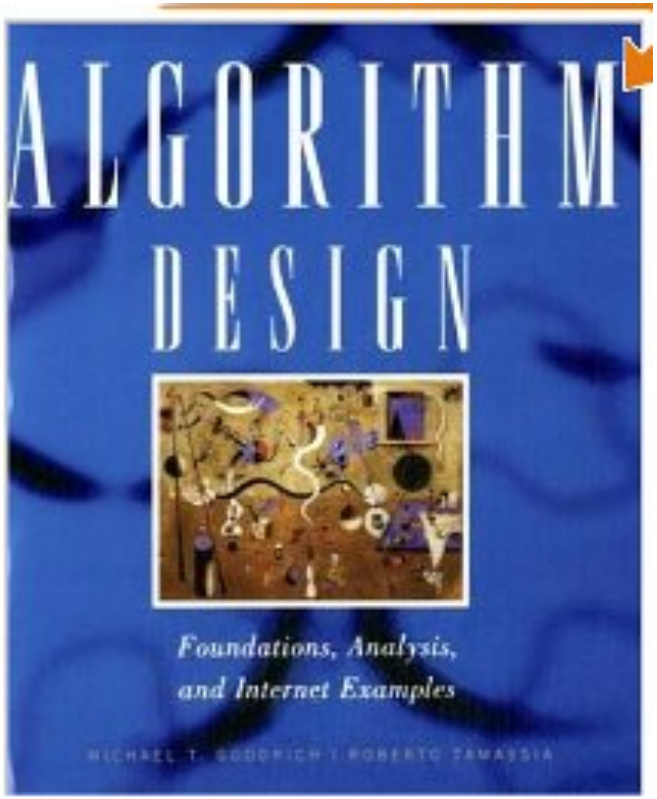


Also known as CLRS book

Introduction to Algorithms, 2nd edition, 2001,

- Cormen, Leiserson, Rivest, and Stein

Reference Book

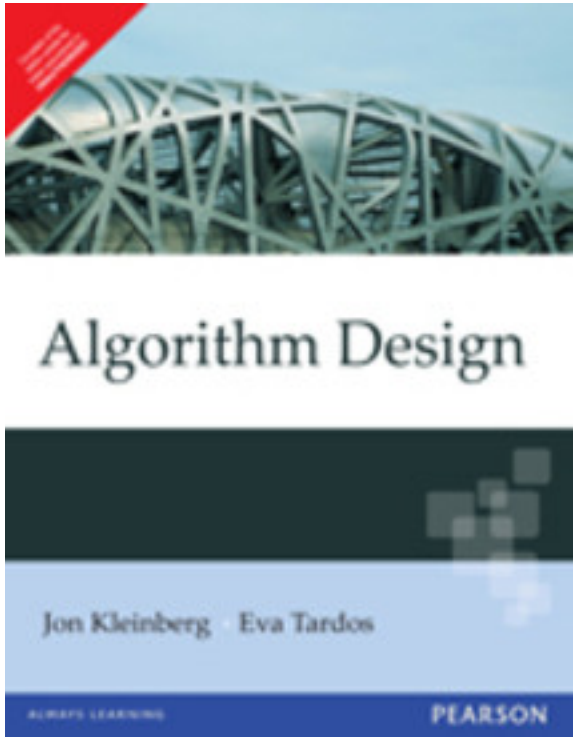


Micheal T. Goodrich and Roberto Tamassia:

Algorithm Design: Foundations, Analysis and Internet examples

(John Wiley &Sons, Inc., 2002)

Reference Books

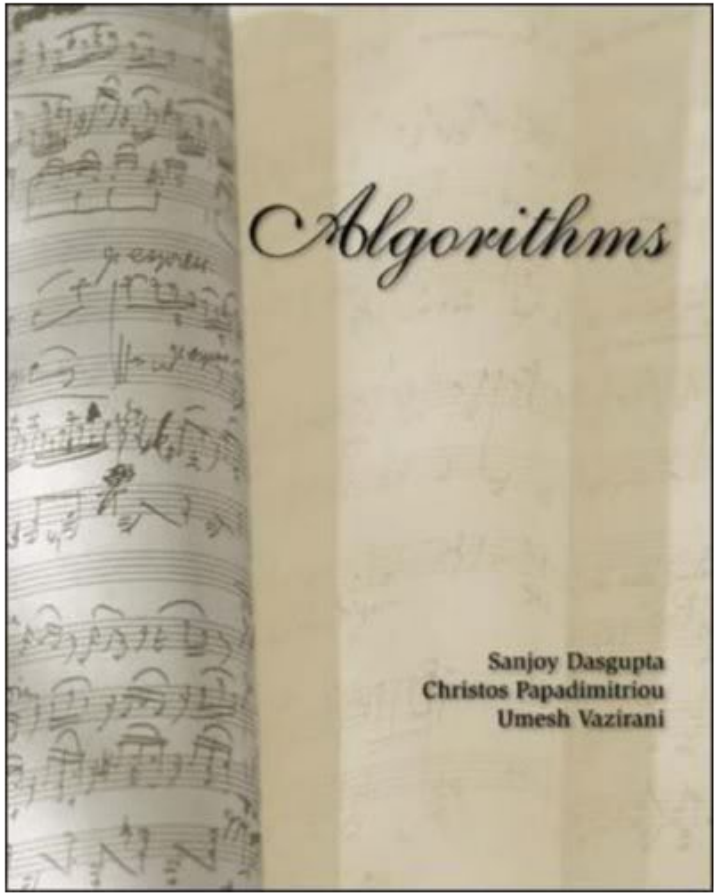


Jon Kleinberg and Eva Tardos. ***Algorithm Design***. Pearson Education. (2007)

Reference Books

Data Structures and Algorithms - Alfred V. Aho, John E. Hopcroft, **Jeffery D.Ulman**

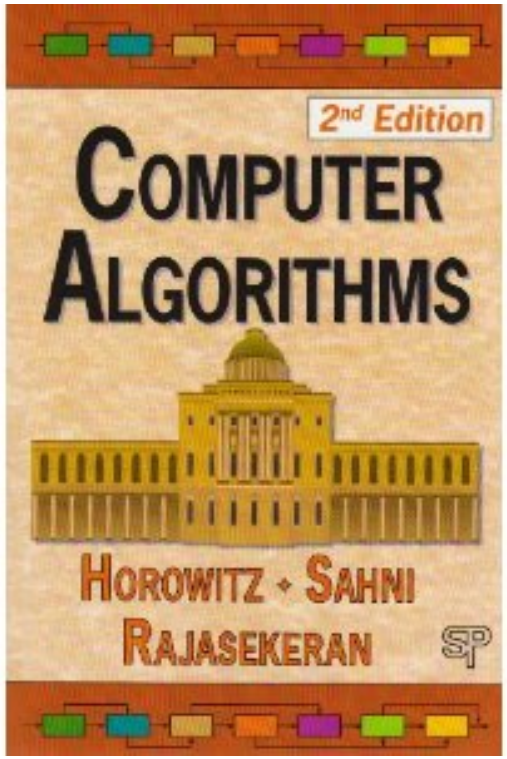
Reference Books



Sanjoy Das Gupta, Christos Papadimitriou, Umesh Vazirani: Algorithms

(Tata McGraw-Hill Publishers)

Reference Books



Ellis Horowitz, Sartaj Sahni, Sanguthevar Rajasekaran. **Computer Algorithms**

Thank You!!