1 First Class

Thursday, January 23, 2025 12:35 PM

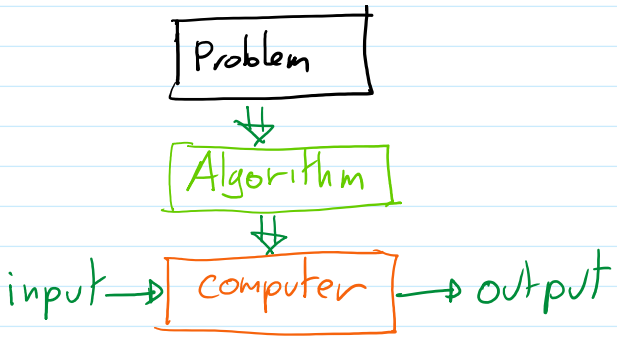
Back to Basics:

Q: what is Computer Science?

"Problems"

¿What problems can be solved in a systematic / automated way, and how?

"Algorithms"



Algorist vs. abacist (1508)

- ALGORITHMS

DEF: An algorithm is a sequence of unambiguous instructions for solving a problem.

or unambiguous instructions for solving a problem.

Solving :- the obtain the required output from the specified input.

- the non-ambiguity of each step cannot be compromised.

- the range of the input must be specified.
- the same algorithm can have multiple representations
- there may be more than one algorithm to solve the same problem.
  - usually based on different ideas
  - usually have different characteristics

E.g. Greatest common Denominator.

gcd(m,n) :- the biggest number that divides m and n.

Let. $m \geq n$

- gcd_v1(m,n)

```
start with t ← n.
while t > 0 do
    if t divides m and t diveds n
        t is the solution
    else
        substract 1 from t
```

- gcd_v2(m,n)

Let m's prime factors be $p_1 * p_2 * p_3 * \ldots * p_i$
Let n's prime factors be $q_1 * q_2 * q_3 * \ldots * q_k$
Multiply the common prime factors.

- gcd_v3(m,n) Euclid's Algorithm.

form 1

1) if n is equal to 0, answer m

form 1

1) if n is equal to 0, answer m
2) divide m by n and assign the reminder to r
3) Assign n to m and r to n
4) repeat from step 1

75 ~ 300 A.D.

form 2

```
while n ≠ 0 do
    r ← m mod n
    m ← n
    n ← r
return m
```

Note on notation

← assignment	=
=	==
	===
and	&&
or	||
≠	!=

- PROCESS OF PROBLEM SOLVING

1) Understand the problem

- assumptions:- "given a list of numbers"
  - integers? reals? negatives?
- spec. of the input
  - "in the list a number appears only once"
  - "all numbers are positive less than 255"
- Abstraction:- the removal of unnecessary information.
  - model the problem as a mathematical object.

2) Decide on Computational Needs and Constraints

- Speed & Memory
- parallel or sequential execution
- exact or approximate solution

3) Specify and Design Algorithm

3) Specify and Design Algorithm

- Algorithm that operates on the mathematical object
- the most important operations of the algorithm are used to decide the appropiate data structures.

4) Show correctness

- Show that the algorithm solves the problem
  - the output will be correct for any valid input when the algorithm terminates.

5) Analysis

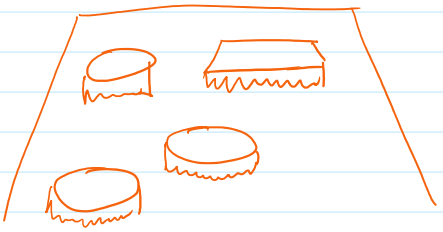
- Resource Utilization: → Time (number of steps)
  → Space (size of memory needed)
- characteristics
  - use of "randomness"
  - "stability"

6) Implement

7) Profit ?!?

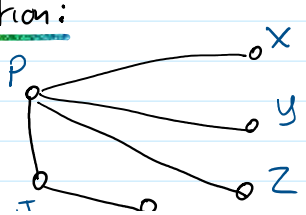
E.G. Problem.

Romeo and Juliet are getting married.
Family members hate each other
Therefore, at the reception, some family members cannot sit together at the same table.
You have a list of guests and for each guest a list of who they hate.
How to sit the guests?
Which tables to assign them?

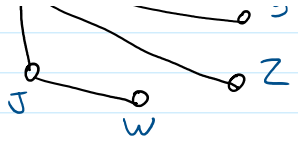


Petro hates X, Y, Z
John hates Petro, W
⋮

1.- Abstraction:



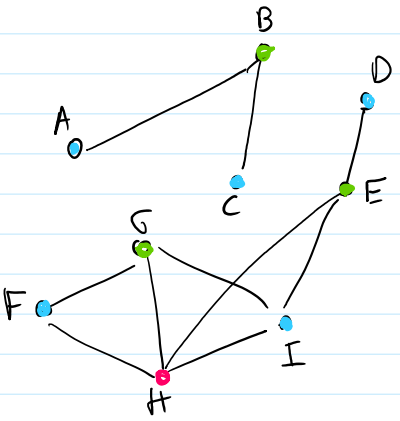
Graph.



Graph.

3.- Algorithm: "Graph partition"

while there are uncolored vertices:
- pick an uncolored vertex v and color it a new color c
- for every other uncolored vertex u
  If u is not connected to a vertex of color c, paint u of c.



{A, C, D, I, F}
{B, G, E}
{H}

Operations: - look for uncolored vertices
- iterate over vertices.
- iterate over vertex neighbors.

D.S.= Adjacency List

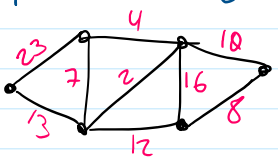
4.- Correctness.-

5.- Analysis.- $|V|^2$ iterations.

6.- Implement.

- IMPORTANT PROBLEM CATEGORIES:
  - Sorting
    re-arranging a list of elements in non-decreasing order
  - Searching
    find an item in a collection.
    - depends on the representation of the collection
  - String Processing
    e.g. - determine whether a string is a palindrome
    - finding a substring - search & replace
    - match a pattern
    - find how different two strings are.
  - Graph problems.

- Graph problems.
  - Partitioning.
  - finding paths
  - finding cycles.
  - adding / removing edges
  - if graph is weighted



★ shortest (lowest-cost) path

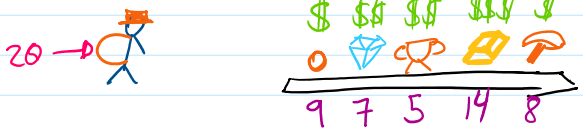
★(TPS) Travelling Salesman Problem

- find a cycle that visits all nodes in the graph of minimum cost.

- Combinatorial Problems.

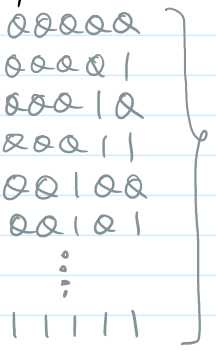
find a combination or permutation of items that satisfies a criteria or/and minimize/maximize a property

★ Knapsack Problem.



What items to choose to maximize loot.

How many candidate selections are there?



$2^5 = 32$

$2^6 = 64$

$2^{16} \approx 65,000$

$2^{32} \approx 4,000,000,000$

These problems are HARD
- no efficient algorithms are known.
- there is no proof that an efficient algorithm does not exists.

¿ P = NP ?

- Geometric Problems.

Problems on points, lines & polygons

e.g. - line intersection

- ☆ closest points.
- line polygon & polygon-polygon intersection.

☆ Convex Hull.

;

; ; ;