W1

▪ ECS 30

— Bernd Hamann
- Instructor

- WELCOME! WE WILL LEARN TOGETHER.

- SYLLABUS

- COURSE WEB PAGES: https://smartsite.ucdavis.edu

  → ~~General Course Materials~~

  → Announcements, Questions & Answers

- TEACHING ASSISTANTS:

  → Responsible for issues concerning written homework, programming projects, exams, grading

- PROGRAMMING:

  → Must run on Kemper Hall computers!

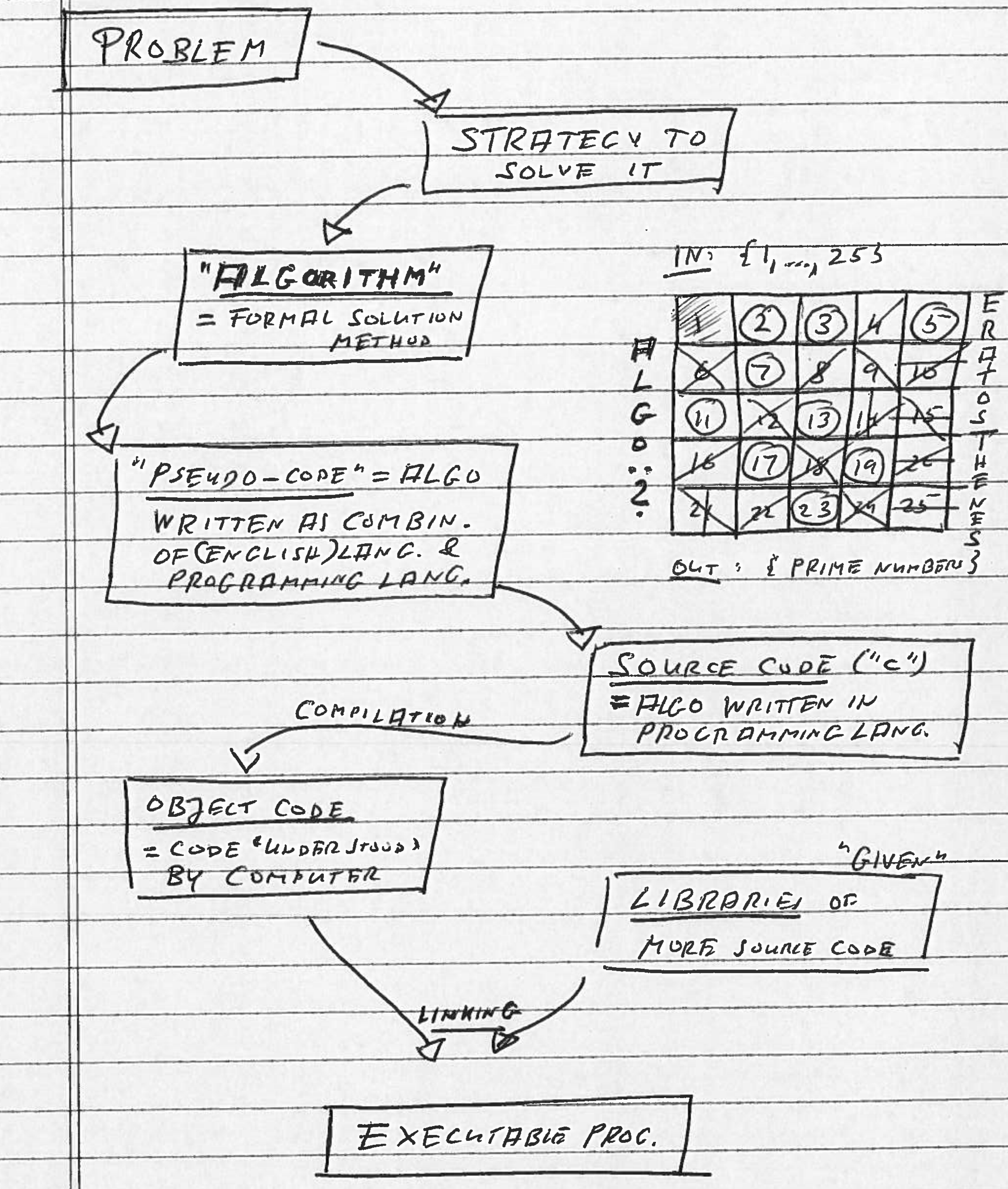
- UNIX/EDITOR/DEBUGGER LABS: TAs organize

QUESTIONS?

- NOTE: 1) Work on assignments continually!
  2) Ask TAs for answers to general questions! Answers on course web pages

- READ CH. 1&2

CHAPTER 1 READ



PROBLEM

STRATEGY TO SOLVE IT

"ALGORITHM" = FORMAL SOLUTION METHOD

"PSEUDO-CODE" = ALGO WRITTEN AS COMBIN. OF (ENGLISH) LANG. & PROGRAMMING LANG.

SOURCE CODE ("C") = ALGO WRITTEN IN PROGRAMMING LANG.

COMPILATION

OBJECT CODE = CODE "UNDERSTOOD" BY COMPUTER

"GIVEN" LIBRARIES OF MORE SOURCE CODE

LINKING

EXECUTABLE PROG.

IN: {1, ..., 25}

ALGO: 2

ERATOSTHENES

~~1~~	2	3	~~4~~	5
~~6~~	7	~~8~~	~~9~~	~~10~~
11	~~12~~	13	~~14~~	~~15~~
~~16~~	17	~~18~~	19	~~20~~
~~21~~	~~22~~	23	~~24~~	~~25~~

OUT: {PRIME NUMBERS}

W 1

ALGO: "FORMAL DESCRIPTION OF STEP-BY-STEP PROBLEM SOLUTION"

DEF.

(i) **UNAMBIGUOUS**

→ All possible cases considered

(ii) **EFFECTIVE/EXECUTABLE**

→ All individual steps of algo can effectively be executed (on a computer)

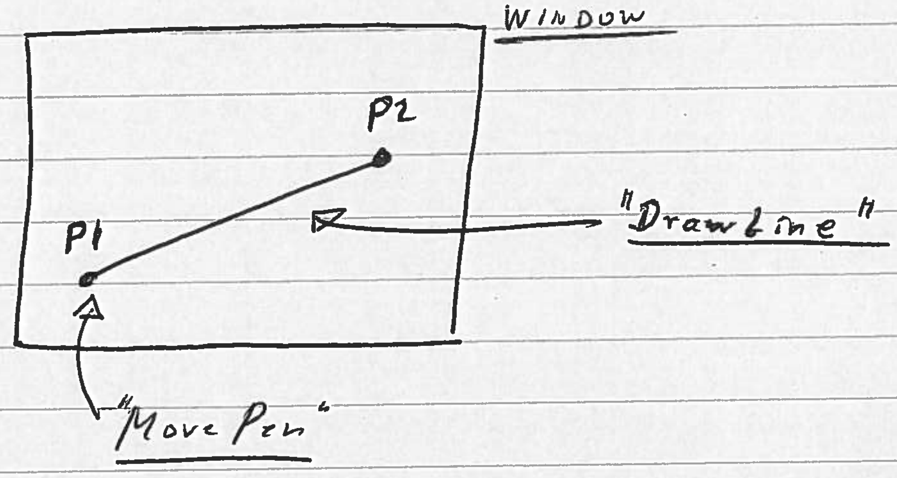
(iii) **TERMINATION** ("must finish")

OVERVIEW

- SIMPLE STATEMENTS   **if**, **for**, **while**, ...
- FUNCTIONS/PROCEDURES

E.g.:   radius → area of circle
2 pnts → drawing of line
string → no. of its characters

- LIBRARY → GRAPHICS LIB.! ☺



W1

- **INTERFACE**
  - → Graphics Lib. "graphics.h"
    - ⇒ Def. of all AVAILABLE graphics fcts.
  - → Math Lib. math.h
    - ⇒ Def. of AVAILABLE math. fcts.

- **STRINGS**
  - → E.g.: COMPILER: IN = Source File in C
    - ⇒ PARSER: parses all text input ...
      - ⇒ ERROR detection ...

- **MODULAR DEVELOPMENT**

E.g.:

"SIMULATE AIR FLOW AROUND AIR CRAFT"

COMPLEX PROBLEM

→ Geometry / CAD: geometry + GUI libraries

→ Numerical Methods: math. libraries for ...

→ Visualization: graphics/vis./GUI libraries

"DECOMPOSITION"

W1

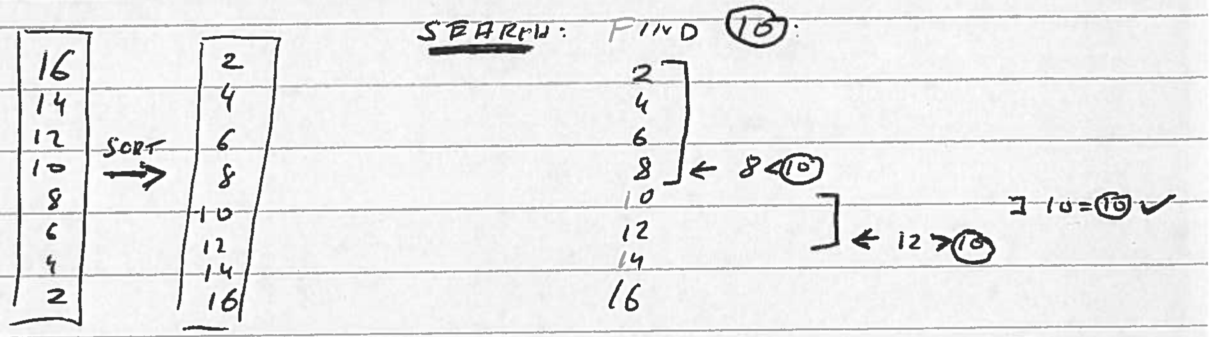
ARRAYS

A[0]				A[N-1]
2	1	111	89	47

→ array of integers

→ use index to access element

SORTING & SEARCHING



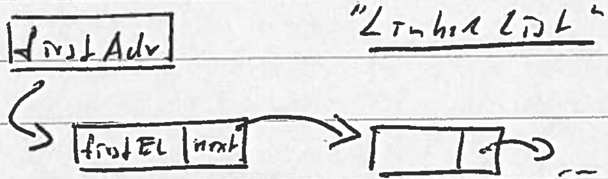
⇒ SEARCHING IN UNSORTED / SORTED LISTS

POINTERS

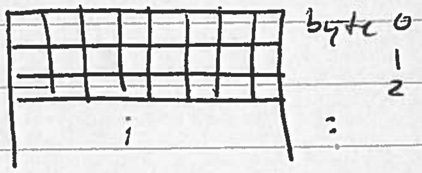
address	name	nextAdr
0	ROGA	/ NIL
1	GUSF	4
2	MART	0
3	LEVI	2
4	HAMA	3

First

"Linked List"



- Memory



FILES / RECORDS / DATABASE

one record →	lastName	firstName	SSN	ann Sal	...
	...	...			

W1

- **RECURSION**

"Reduce a complex problem to a simpler one"

$$\rightarrow a \cdot b = \begin{cases} 0, & \text{if } b = 0 \\ a + (a \cdot (b-1)), & \text{otherwise} \end{cases}$$

EX: $\underset{(a)}{3} \cdot \underset{(b)}{2} = 3 + (3 \cdot 1)$

$= 3 + (3 + (3 \cdot 0))$

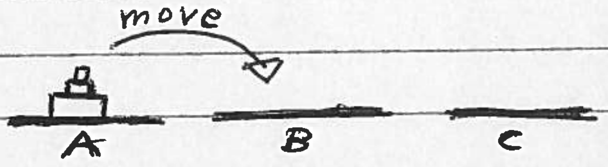
$= 3 + (3 + 0)$

$= 3 + 3$

$= \underline{\underline{6}}$

Reduce '·' to just '+' and '-'

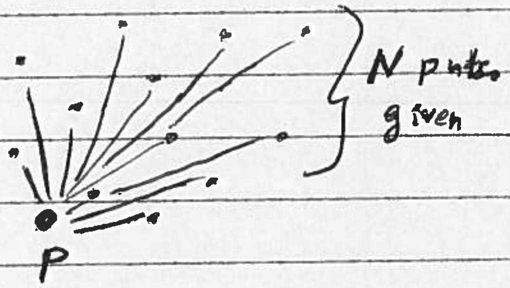
→ Tower of Hanoi: ??? Explain!



"Move tower from A to B"

- **COMPLEXITY**

e.g., determine point closest to P



"Check distances to all N pnts"

⇒ Linear complexity

"O(N)"

or: WHICH POINT PAIR (p_i, p_j) HAS MINIMAL DISTANCE?

⇒ $O(N^2)$???

BEST METHOD ?