CSE 101-01 4-11-23

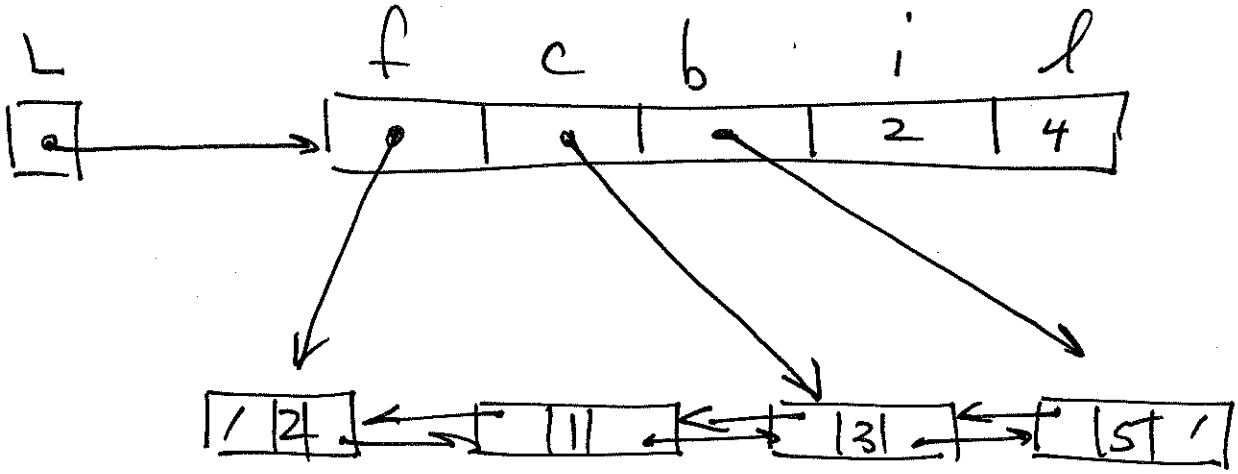
---

Pa1 : List ADT

Ex.

client view : ( 2 1 3 5 )
0 1 2 3

inside view :

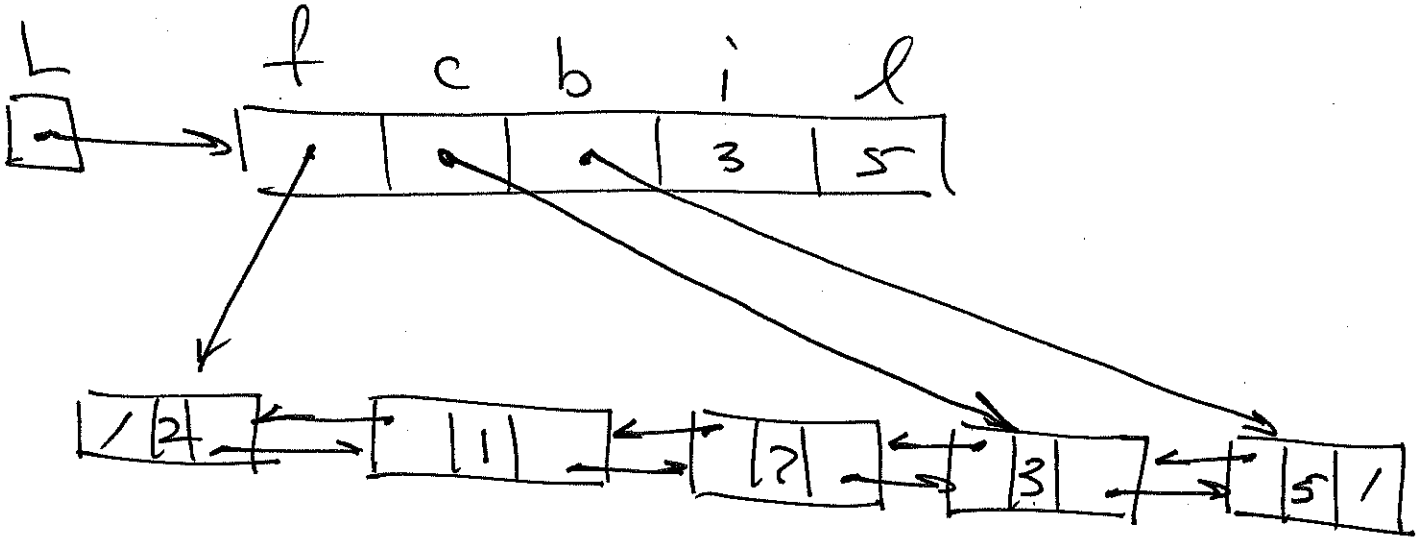



InsertBefore(L, 7)

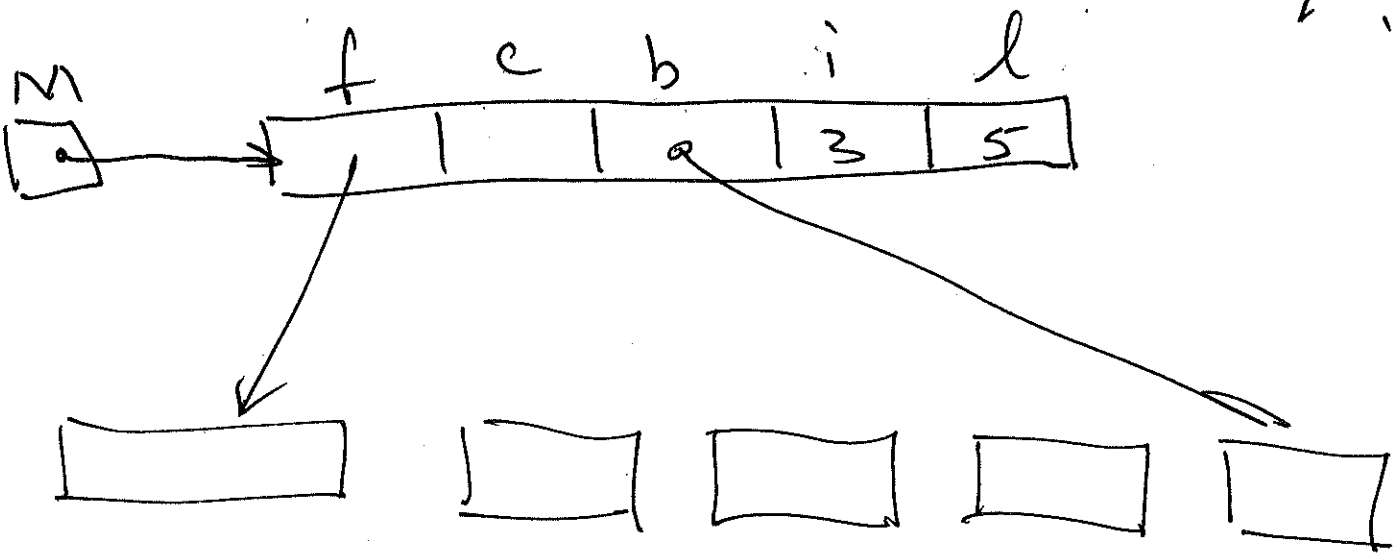
0 1 2 3 4

client view: ( 2 1 7 3 5 )

inside view:




copyList(): makes a deep copy!



Pair: example

0 1 2 3

A = ( "c" "a" "b" "d" )

want: L = ( 1 2 0 3 )

| | | sub-array |
|---|---|---|
| start: | L = ( ) | ( ) |
| insert 0: | L = ( 0 ) | ( "c" ) |
| insert 1: | L = ( 0 ) | |
| | L = ( 1 0 ) | ( a c ) |
| insert 2: | L = ( 1 0 ) | |
| | L = ( 1 0 ) | |
| | L = ( 1 2 0 ) | ( a b c ) |

insert 3: L = (1 2 0)

L = (1 2 0)

L = (1 2 0)

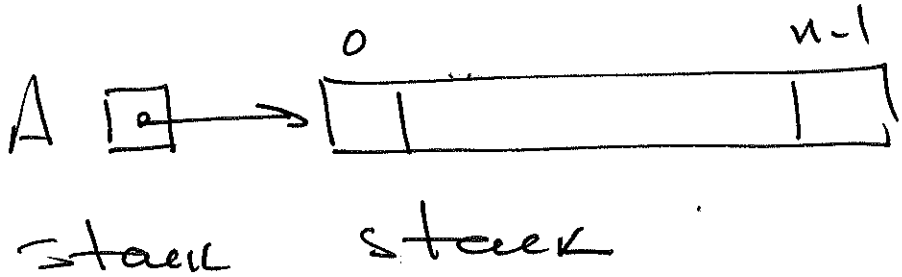
L = (1 2 0)

L = (1 2 0 3)

## Variable length arrays (VLA) in C

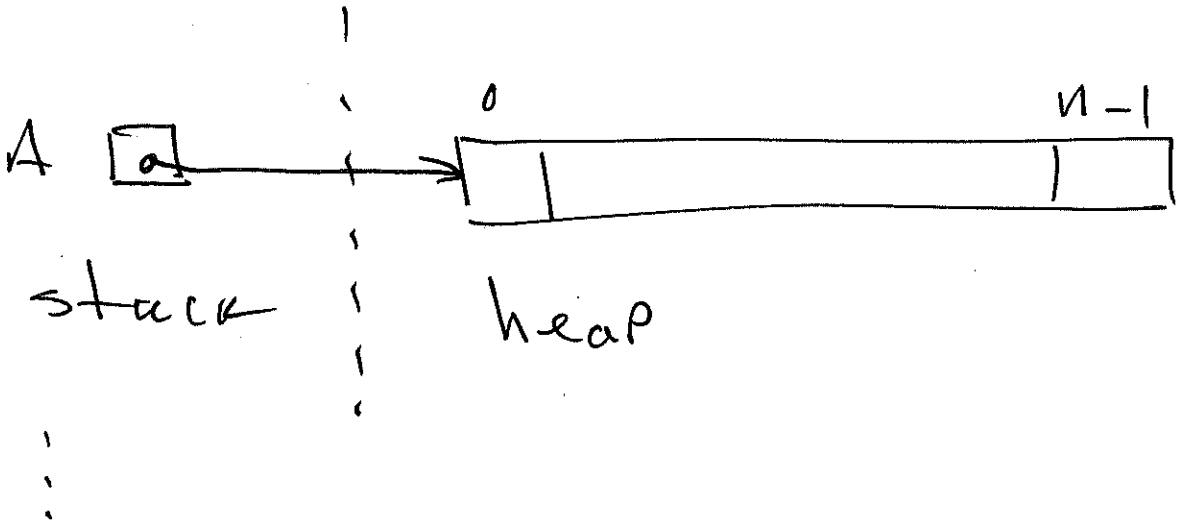
In ISO C99 the following is possible

```
int n; // get n from user
int A[n]; <-- array in stack!
for(int i=0; i<n; i++) A[i] = something;
```




Don't use VLA's. Instead use heap memory.

```
int * A = calloc(n, sizeof(int));
```




```
free(A);
A = NULL;
```

# Graphs

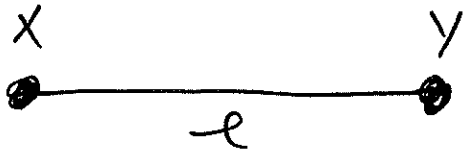
**Defn** A **graph** is a pair of sets

$$G = (V, E)$$

where

- $V \neq \emptyset$ (vertex)
- $E \subseteq V^{(2)} = \{2\text{-sets of } V\}$

i.e. $e \in E$ is an unordered pair of vertices: $e = \{x, y\}$ with $x, y \in V$

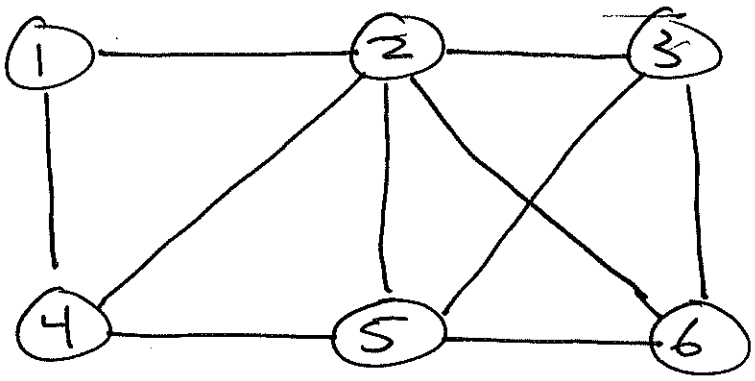



$$= \{x, y\} = xy$$
$$= \{y, x\} = yx$$

Ex.

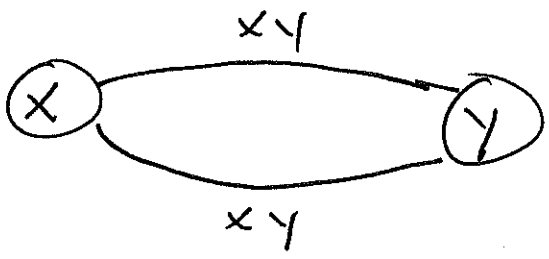
$V(G) = \{1, 2, 3, 4, 5, 6\}$

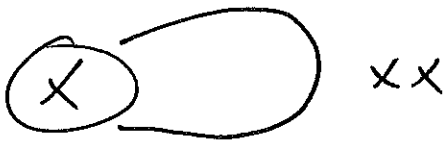
$E(G) = \{12, 14, 23, 24, 25, 26, 35, 36, 45, 56\}$




- 1 is <u>adjacent</u> to 2
- 12 is <u>adjacent</u> to 23
- 1 is <u>incident</u> with 14
- the <u>ends</u> of 56 are 5 and 6
- 56 <u>joins</u> 5 to 6

impossible




$u - v$

Defn A Walk in $G$ is a sequence

$$u = x_0, x_1, x_2, \ldots, x_{k-1}, x_k = v$$

of vertices where each consecutive pair $\{x_i, x_{i+1}\}$ $(0 \le i \le k-1)$ are adjacent

a 1-6 walk: 1, 2, 4, 5, 2, 3, 5, 6

A trail is a walk in which no edge is repeated. a Path is a trail in which no vertex is repeated.

Ex 1-6 Path: 1, 2, 4, 5, 3, 6

length = 5