Binary Search:

Suppose DATA is an array which is sorted in ***INCREASING ORDER***, or equivalently, alphabetically.

Then, *Binary Search algorithm* is an extremely efficient searching algorithm to find the location LOC of a given ITEM of information in DATA.

Binary search algorithm applied to array DATA works as follows:

DATA[BEG], DATA[BEG+1], DATA[BEG+2],........, DATA[END]

This algorithm compares ITEM with the middle element DATA [MID] of the segment, where MID is obtained by

MID=INT((BEG+END)/2)

✓ If DATA[MID]=ITEM, then the search is **SUCCESSFUL**.

We set LOC:=MID

....Otherwise a new segment of DATA is obtained.

Binary Search:

(a) If ITEM < DATA[MID], then ITEM can appear only in the left half of the segment:

DATA[BEG], DATA[BEG+1], ….DATA[MID-1]

So, we resent END:=MID-1 and begin searching again.

(b) If ITEM> DATA[MID], then ITEM appear only in the right half of the segment:

DATA[MID+1], DATA[MID+2],..........DATA[END]

So, we reset BEG:=MID+1 and begin searching.

- ✓ Initially, we begin with entire array DATA, i.e. We begin wit BEG=1 and END=n, or more generally, with BEG=LB and END=UB.

- ✓ If ITEM is not in DATA, then eventually we obtain

END< BEG

Which means the search is **Unsuccessful**

So, SET LOC:=NULL (OUT side of DATA indices)

Binary Search: : Example

Searching the array below for the value **42**:

											10						
	-4	2	7	10	15	20	22	25	30	36	42	50	56	68	85	92	103

min mid max

© Dr. Md. Golam Rashed, Assoc. Professor, Dept. of ICE, RU

Binary Search: Algorithm

BINARY (DATA, LB, UB, ITEM, LOC)

(This algorithm finds the location LOC of item in DATA or sets LOC:=NULL)

1. [Initialize segment variables.]
   Set BEG:=LB, END:=UB, and MID=INT((BEG+END)/2).
2. Repeat Steps 3 and 4 while BEG<=END and DATA[MID] $\neq$ ITEM
3. If ITEM<DATA[MID], then
   Set END:=MID-1.
   Else:
   Set BEG:=MID+1 [End of If structure.]
4. Set MID:=INT((BEG+END)/2
   [End of Step 2. loop]
5. If DATA[MID]=ITEM, then:
   Set LOC:=MID
   Else:
   Set LOC:=NULL. [End of If structure]
6. Exit

© Dr. Md. Golam Rashed, Assoc. Professor, Dept. of ICE, RU

Binary Search: Limitations

Your Task

© Dr. Md. Golam Rashed, Assoc. Professor, Dept. of ICE, RU

Linear Arrays/ One Dimensional Array

- In general, the length or the number of data elements of the array can be obtained from the index set by the formula.

$$\text{Length}=\text{UB}-\text{LB}+1$$

where UB is the largest index, called the upper bound, and LB is the smallest index, called the lower bound of the array.

✓ Using the base address of a array LA, the computer calculates the address of any element of LA by the following formula:

$$\text{LOC}(\text{LA}[K])=\textit{Base}(\text{LA})+w(K-\text{lower bound})$$

w-is the number of words per memory cell for the array LA.

Two Dimensional Array

- A two dimensional $m \times n$ array A is a collection of $m.n$ data elements such that each element is specified by a pair of integer, called subscripts(J,K), with the property that…

$$1 \leq J \leq m \text{ and } 1 \leq K \leq n$$

- A two dimensional arrays are called **metrices** in mathematics and **tables** in business application; hence two-dimensional arrays are sometimes called *matrix arrays*.

© Dr. Md. Golam Rashed, Assoc. Professor, Dept. of ICE, RU

Two Dimensional Array: Representation in Memory

Let A be a two-dimensional array $m \times n$ array.

Although A is pictured as a rectangular array of elements with m rows and n columns,

Student	Test 1	Test 2	Test 3	Test 4
1	84	73	88	81
2	95	100	88	96
3	72	66	77	72
⋮	⋮	⋮	⋮	⋮
25	78	82	70	85

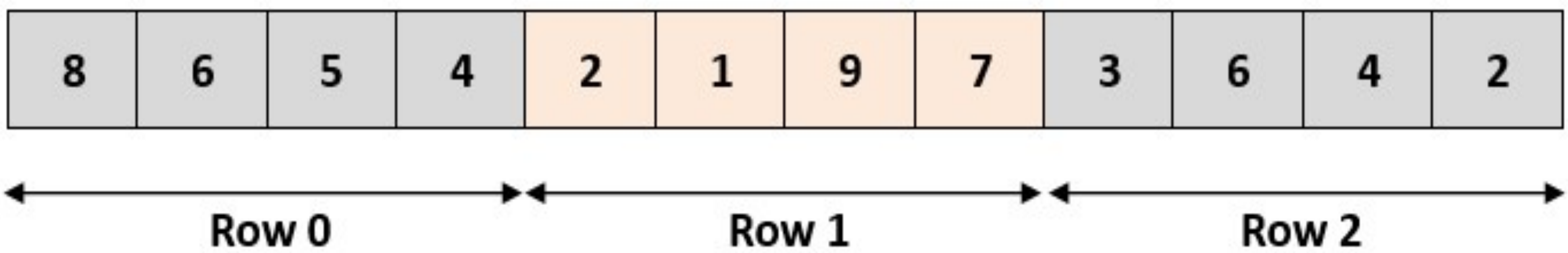
The array will be represented by a block of $m.n$ sequential memory location.

Two Dimensional Array: Representation in Memory

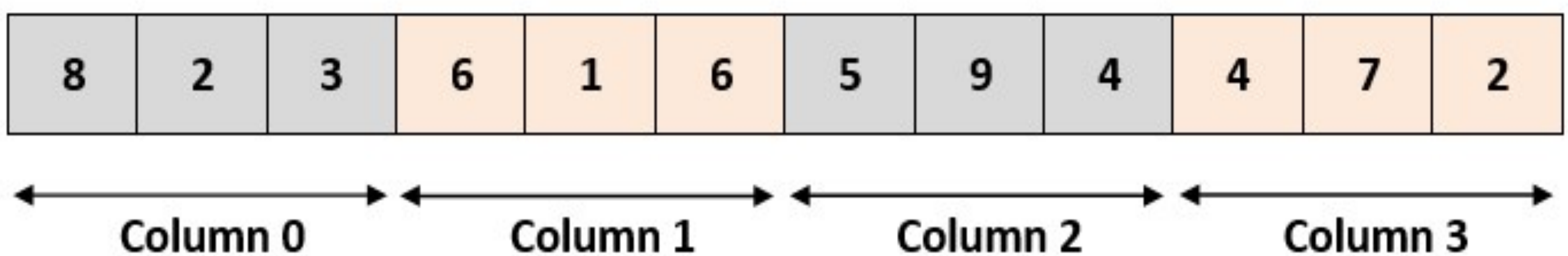
The programming language will store the array A either

- ➢ Column by Column – called *Column Major Order (CMO)*
- ➢ Row by Row-called Row Major Order(RMO)

Row-Major (Row Wise Arrangement)



Column-Major (Column Wise Arrangement)



Two Dimensional Array: Accessing Element

For one-dimensional array, the computer uses the formula

LOC(LA[K]) = *Base*(LA) + w(K-1)

to find the address of LA[K] in time independent of K, where w is the number of words per memory cell for the array LA and 1 is the lower bound of the index set of LA.

A similar situation also holds for any two-dimensional $m \times n$ array A. Computer keep track of *Base*(A)-the address of the first element A[1,1] of A and compute the address LOC(A[J,K]) of A[J,K] using the formula:

- for CMO, LOC(A[J,K])=*Base*(A)+w[M(K-1)+(J-1)]
- For RMO, LOC(A[J,K])=*Base*(A)+w[N(J-1)+(K-1)]

Tech Yourself: Example 4.12,

Two Dimensional Array: Problem

- Given: 3×4 Integer matrix with base address 1000.
- Find out the location of A[3][2].

Using Row major Formula

- The Location of element A[i, j] can be obtained by evaluating expression:
  LOC (A [i, j]) = Base_Address + W [M (i) + (j)]
  Here,
  Base_Address is the address of first element in the array.
  W is the word size. It means number of bytes occupied by each element.
  N is number of rows in array.
  M is number of columns in array.

Two Dimensional Array: Problem

- Given: 3×4 Integer matrix with base address 1000.
- Find out the location of A[3][2].

Using Row major Formula

LOC (A [i, j]) = Base_Address + W [M (i) + (j)]

- Base (A) : 1000
- w : 2 (because an integer takes 2 bytes in memory)
- N : 4
- J : 3
- K : 2
- Now put these values in the given formula as below:
- LOC (A [3, 2]) = 1000 + 2 [4 (3-1) + (2-1)]
- = 1000 + 2 [4 (2) + 1]
- = 1000 + 2 [8 + 1]
- = 1000 + 2 [9]
- = 1000 + 18
- = 1018

Two Dimensional Array: Problem

- Given: 3×4 Integer matrix with base address 1000.
- Find out the location of A[3][2].

Using Column major Formula

- LOC (A [J, K]) = Base (A) + w [M (K-1) + (J-1)]
- Here
- LOC (A [J, K]) : is the location of the element in the Jth row and Kth column.
- Base (A) : is the base address of the array A.
- w : is the number of bytes required to store single element of the array A.
- M : is the total number of rows in the array.
- J : is the row number of the element.
- K : is the column number of the element.

Two Dimensional Array: Problem

ICE 2261

- Given: 3×4 Integer matrix with base address 1000.
- Find out the location of A[3][2].

Using Column major Formula

- Base (A) : 1000
- w : 2 (because an integer takes 2 bytes in memory)
- N : 4
- J : 3
- K : 2

- LOC (A [3, 2]) = 1000 + 2 [3 (2-1) + (3-1)]
- = 1000 + 2 [3 (1) + 2]
- = 1000 + 2 [3 + 2]
- = 1000 + 2 [5]
- = 1000 + 10
- = 1010

- LOC (A [J, K]) = Base (A) + w [M (K-1) + (J-1)]

Pointers

Let DATA be any array. A variable **P** is called a *pointer* if **P** “points” to an element in DATA, i.e., if **P** contains the address of an element in DATA.

Pointer Arrays

An array **PTR** is called a *pointer array* if each element of **PTR** is a pointer

Pointer and Pointer array are used to facilitate the processing the information in DATA

Group 1	Group 2	Group 3	Group 4
Evans	Conrad	Davis	Baker
Harris	Felt	Segal	Cooper
Lewis	Glass		Ford
Shaw	Hill		Gray
	King		Jones
	Penn		Reed
	Silver		
	Troy		
	Wagner		

How the membership list can be stored in memory keeping track of the different groups?

© Dr. Md. Golam Rashed, Assoc. Professor, Dept. of ICE, RU

Possible Solutions to keep in memory

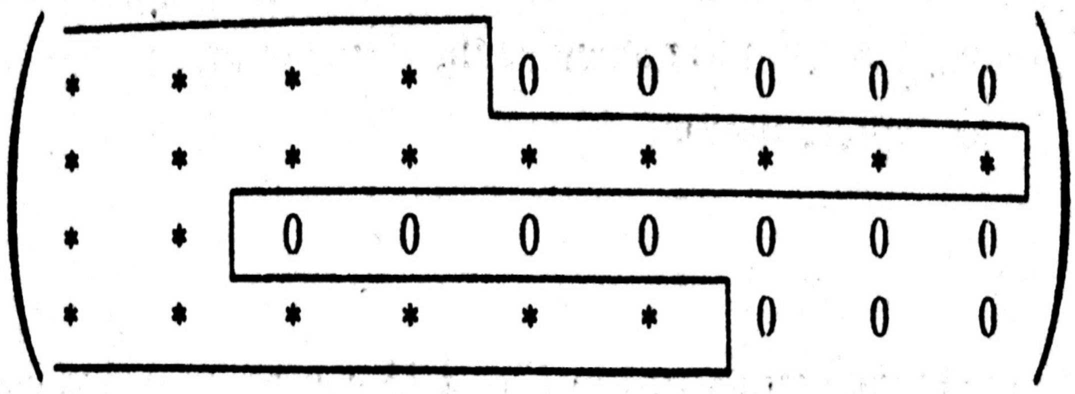
- Possible solutions: using
  - 2D $4 \times n$ array where each row contain a group, or
  - 2D $n \times 4$ array where each column contains a group.
- These structure allows us to access each individual group, much space will be wasted when the groups vary greatly in size.

Group 1	Group 2	Group 3	Group 4
Evans	Conrad	Davis	Baker
Harris	Felt	Segal	Cooper
Lewis	Glass		Ford
Shaw	Hill		Gray
	King		Jones
	Penn		Reed
	Silver		
	Troy		
	Wagner		

- Here the data will require at least a 36-element 4 x 9 or 9 x 4 arrays to store the 21 names, which is almost twice the space that is necessary.

Representation of 4 x 9 array

Group 1	Group 2	Group 3	Group 4
Evans	Conrad	Davis	Baker
Harris	Felt	Segal	Cooper
Lewis	Glass		Ford
Shaw	Hill		Gray
	King		Jones
	Penn		Reed
	Silver		
	Troy		
	Wagner		



Jagged array

- **Arrays whose rows –or column- begin with different numbers of data elements and each with unused storage locations are said to be *jagged*.**

© Dr. Md. Golam Rashed, Assoc. Professor, Dept. of ICE, RU

Possible Solutions to keep in memory

One group after another

	MEMBER	
1	Evans	Group-1
2	Harris	
3	Lewis	
4	Shaw	
5	Conrad	Group-2
.		
.		
13	Wagner	
14	Davis	Group -3
15	Segal	
16	Baker	Group-4
.		
.		
20	Jones	
21	Reed	

- ✓ Space-Efficient
- ✓ Entire list can be easily processed
- ✓ One can easily print all the names on the list.

- ✓ There is no way to access any particular group.
- ✓ There is no way to find and print only the names in the third group.

© Dr. Md. Golam Rashed, Assoc. Professor, Dept. of ICE, RU

Possible Solutions to keep in memory

	MEMBER	
1	Evans	Group-1
2	Harris	
3	Lewis	
4	Shaw	
5	$$$	
6	Conrad	Group-2
.		
.		
14	Wagner	
15	$$$	
16	Davis	Group-3
17	Segal	
18	$$$	
19	Baker	Group-4
.		
.		
20	Jones	
21	Reed	
25	$$$	

- ✓ Uses only a few extra memory cells- one for each group.
- ✓ Any one now find those names in he third group by locating those names which appear after the second sentinel

- ✓ The list still must be traversed from the beginning in order to recognize the third group.
- ✓ The different groups are not indexed with this representation.

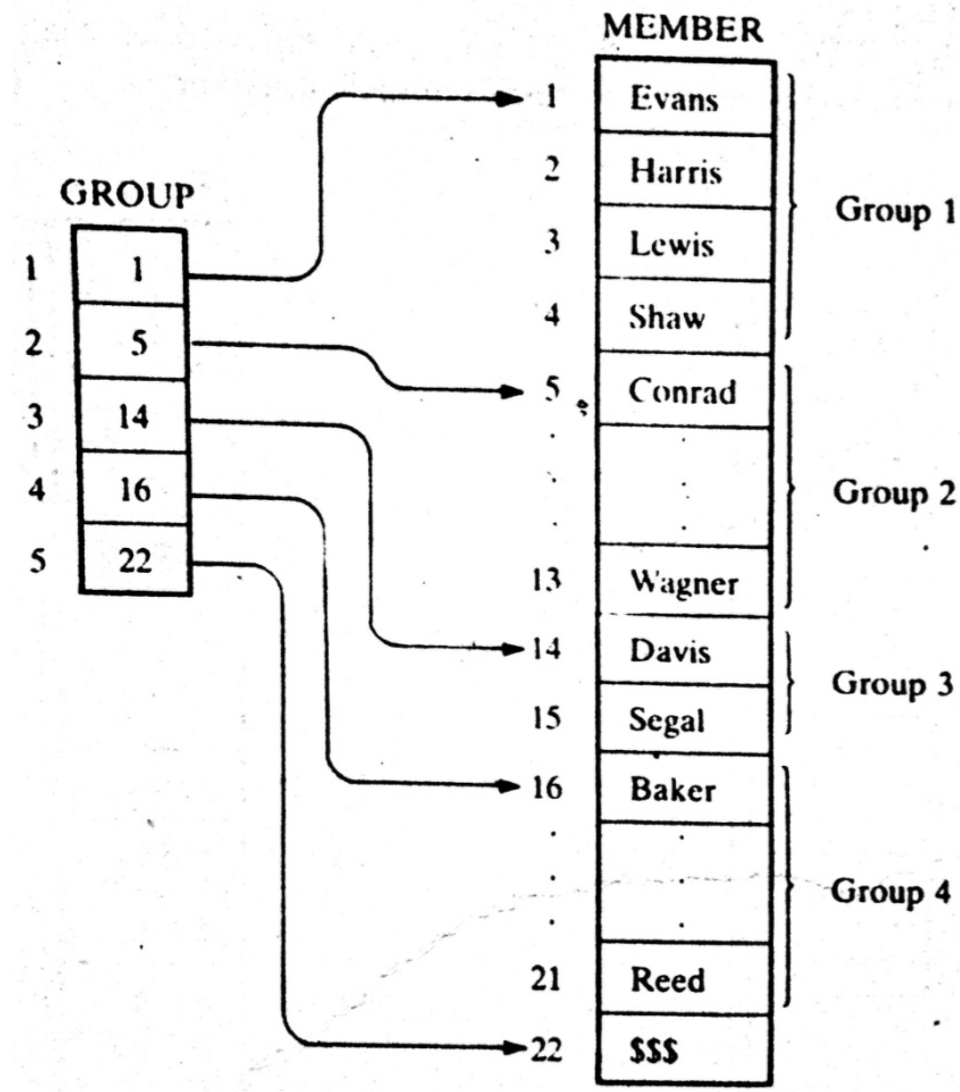
© Dr. Md. Golam Rashed, Assoc. Professor, Dept. of ICE, RU

POINTER ARRAYS

Pointer arrays is introduced in the last two space-efficient data structure.

The pointer array contains the locations of the.....

- ✓ Different groups, or
- ✓ First element in the different groups.
- ✓ GROUP[L] and GROUP[L+1]-1 contain respectively, the first and last element in group L.



- Suppose L=3
- 1st Element of grp 3?
- GROUP[L]=GROUP[3]
  = 14
  = Davis

- Last Element of grp 3?

Pls Try

© Dr. Md. Golam Rashed, Assoc. Professor, Dept. of ICE, RU

POINTER ARRAYS: Example

Last element of grp 3

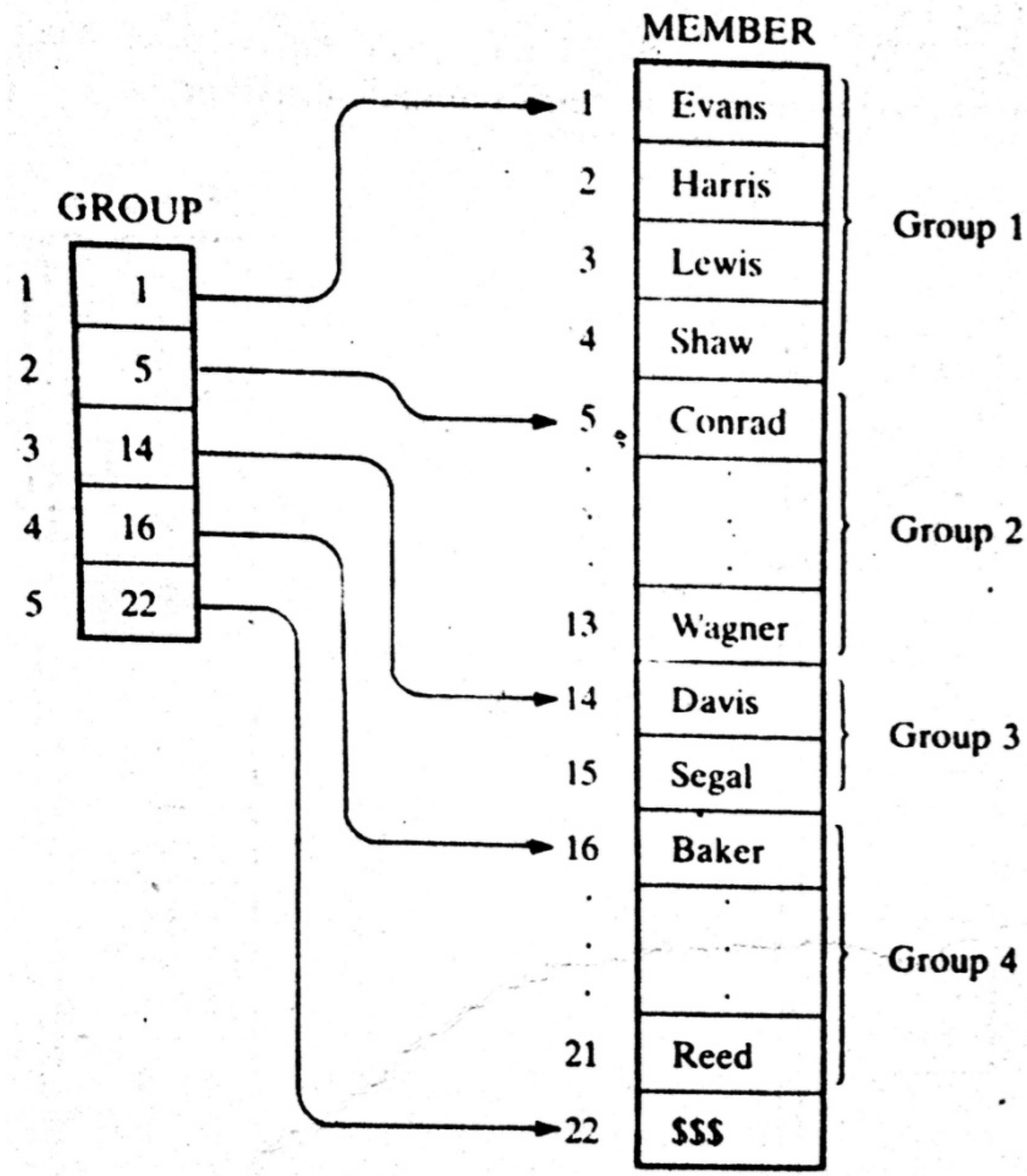
GROUP[L+1]-1

=GROUP[3+1]-1

=16-1

=15

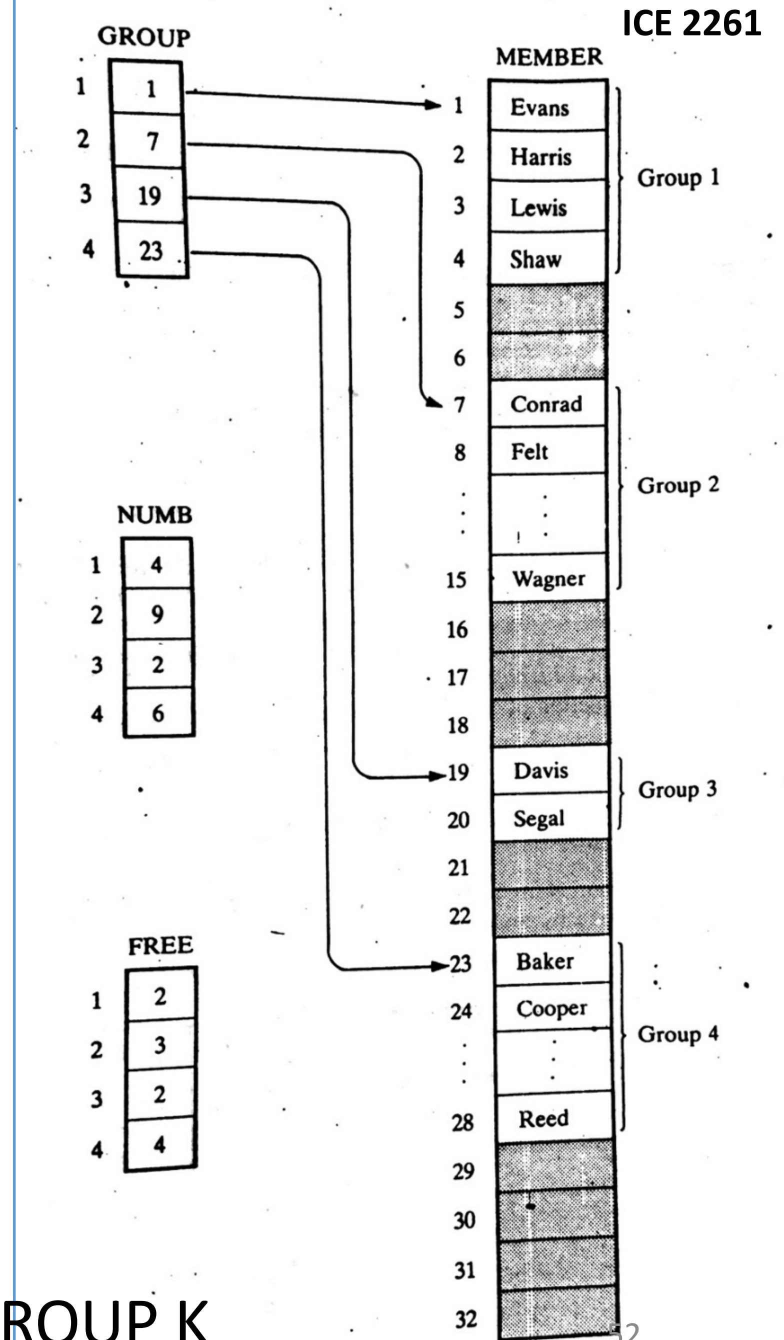
= Segel



© Dr. Md. Golam Rashed, Assoc. Professor, Dept. of ICE, RU

POINTER ARRAYS: Extended

- Here unused memory cells are indicated by the shading.
- Observe that now there are some empty cells between the groups.
- Accordingly, a new element may be inserted in a new group without necessarily moving the elements in any other group.
- Using the data structure, one requires an array NUMB which gives the number of elements in each group.
- Observe that GROUP[K+1]-GROUP[K] is the total number of space available for group K. Hence



FREE[K]=GROUP[K+1]-GROUP[K]-NUMB[K]

Gives the number of empty cells following GROUP K

POINTER ARRAYS: Extended, Example

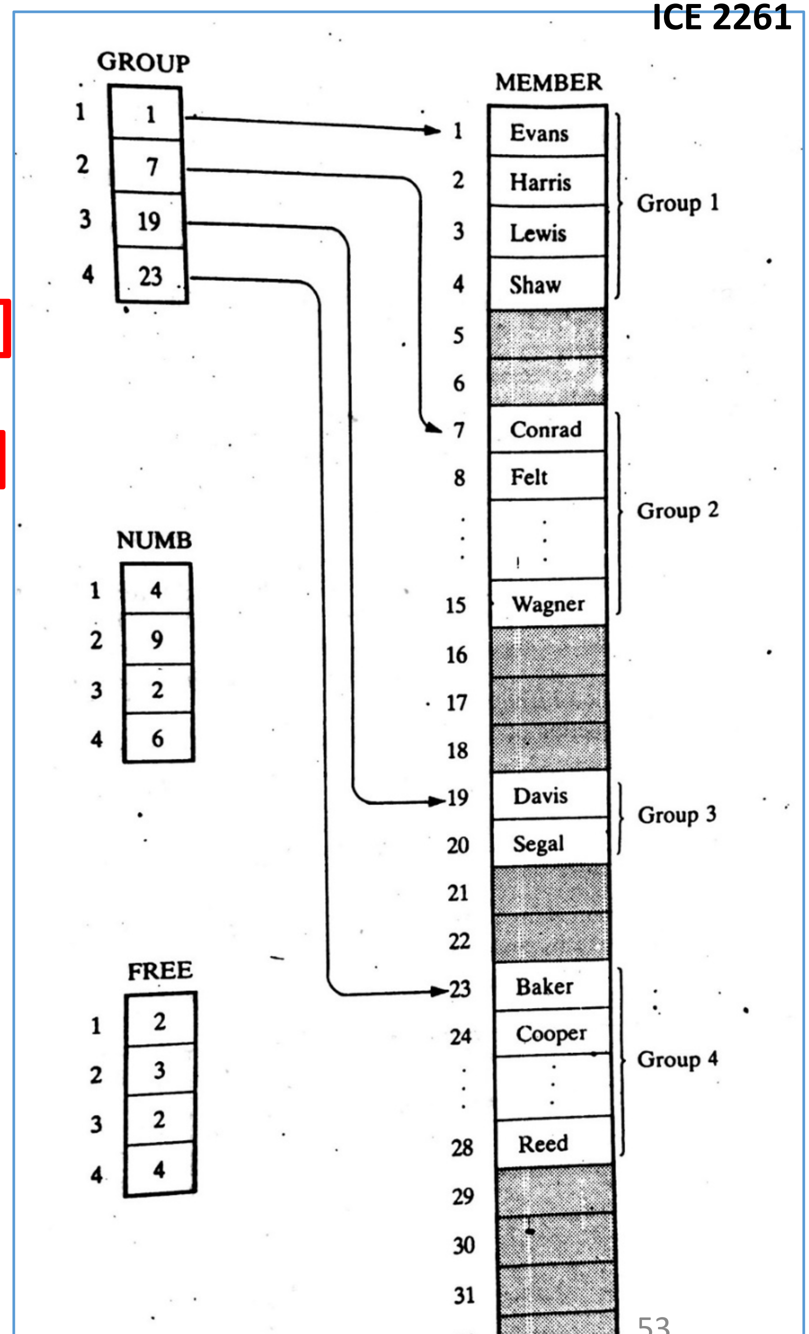
Suppose, we want to print only the number of FREE cells of GROUP 2. Then

FREE[K]=GROUP[K+1]-GROUP[K]-NUMB[K]

FREE[2]=GROUP[2+1]-GROUP[2]-NUMB[2]

= 19-7-9

=3



For GROUP 3?

Try now

© Dr. Md. Golam Rashed, Assoc. Professor, Dept. of ICE, RU

RECORDS

- ✓ A ***record*** is a collection of related data items, each of which is called a field or attribute, and
- ✓ a ***file*** is a collection or similar records.
- ✓ Although, a ***record*** is a collection of data items, it differs from a linear array in the following ways..............
  - ➢ A record may be a collection of nonhomogeneous data;
  - ➢ The data items in a record are indexed by attribute names, so there may not be a natural ordering of its elements.

© Dr. Md. Golam Rashed, Assoc. Professor, Dept. of ICE, RU

RECORDS: Structure Example

1. Newborn
    2. Name
    2. Sex
    2. Birthday
        3. Month
        3. Day
        3. Year
    2. Father
        3. Name
        3. Age
    2. Mother
        3. Name
        3. Age

Under the relationship of group item to sub- item, the data items in a record form a hierarchical structure which can be described by mean of “Level” numbers

Name	Sex	Birthday			Father		Mother	
					Name	Age	Name	Age

© Dr. Md. Golam Rashed, Assoc. Professor, Dept. of ICE, RU

Indexing Items in a Record

- ✓ Suppose we want to access some data item in a record.
- ✓ We can not simply write the data name of the item since the same may appear in different places in the record. For example.....

1. Newborn
    2. Name
    2. Sex
    2. Birthday
        3. Month
        3. Day
        3. Year
    2. Father
        3. Name
        3. Age
    2. Mother
        3. Name
        3. Age

➢ In order to specify a particular item,

- ❖ we may have to *qualify* the name by using appropriate group item names in the structure.
- ❖ This *qualification* is indicated by using decimal points (periods) to separate group items from subitems.
- ❖ Example: Newborn.Father.Age or Father.Age

Fully qualified reference

© Dr. Md. Golam Rashed, Assoc. Professor, Dept. of ICE, RU

Indexing Items in a Record

1. Newborn	1. Newborn(20)
2. Name	2. Name
2. Sex	2. Sex
2. Birthday	2. Birthday
3. Month	3. Month
3. Day	3. Day
3. Year	3. Year
2. Father	2. Father
3. Name	3. Name
3. Age	3. Age
2. Mother	2. Mother
3. Name	3. Name
3. Age	3. Age

Newborn is defined to be a file with 20 records

✓ The Name of the sixth newborn to be referenced by writing ....

Newborn. Name[6]

✓ The age of the father of the 6th newborn may be referenced by writing.....

Newborn.Father.Age[6]

Representation of RECORDS in memory

Since records may contain nonhomogeneous data, the element of a record can not be stored in an array.

See Example: 4.18, 4.20, 4.21

© Dr. Md. Golam Rashed, Assoc. Professor, Dept. of ICE, RU

Teach Yourself with example

Try to understand the SOLVED Problems

NSTART

RSTART

	NAME	ID	NLINK	RLINK
1	Moshi	650		
2	Sakib	422		
3	Riad	704		
4	Liton	462		
5				
6				

NSTART

RSTART

	NAME	ID	NLINK	RLINK
1	Moshi	650		
2	Sakib	422		
3	Riad	704		
4	Liton	462		
5	Mash	632		
6	Fizz	550		