INSTITUTE OF INFORMATION TECHNOLOGY

Jahangirnagar University

জাহাঙ্গীরনগর বিশ্ববিদ্যালয়

IT-1201: Data Structures

for

1st Year 2nd Semester of B.Sc (Honors) in IT (5th Batch)

Lecture: 02

Arrays

Prepared by:

K M Akkas Ali

akkas_khan@yahoo.com, akkas@juniv.edu

Assistant Professor

Institute of Information Technology (IIT)

Jahangirnagar University, Dhaka-1342

Objectives of this Lecture:

- To define array
- To list properties of an array
- Notations used in array
- Declaring an array
- To Represent linear arrays in memory

Prepared by: K M Akkas Ali, Assistant Professor, IIT, JU

Data Structure: Array

What is Array?

- Array is a sequenced list of finite data elements of the same type. Elements of the array are stored respectively in successive memory locations.
- For example, we can use an array named ***salary*** to represent a set of salaries of a group of employees in an organization.
- Since arrays are usually easy to traverse, search and sort, they are frequently used to store relatively permanent collections of data. On the other hand, if the size of the structure and data in that structure are constantly changing, then the array may not be as useful as linked list.
- Some examples where the concepts of an array can be used:
  - ❑ List of temperatures recorded every hour in a day, or a month, or a year.
  - ❑ List of employees in an organization.
  - ❑ List of products and their cost sold by a store.
  - ❑ Test scores of a class of students.
  - ❑ List of customer and their telephone numbers.
  - ❑ Table of daily rainfall.

Prepared by: K M Akkas Ali, Assistant Professor, IIT, JU

Types of Array:

1. Linear Array or One-dimensional Array
2. Multi-dimensional Array (Such as Two-dimensional Array)

Data Structure: Array

One-dimensional or Linear Array:

- By a linear array, we mean a list of ***n*** data elements of similar type where each element is referenced by one subscript. i.e. by a set of ***n*** consecutive numbers called ***index set or subscripts*** , usually 1, 2, 3, ….,n (if unit-based indexing is used) or 0, 1, 2, ……n-1 (if zero-based indexing is used). Here the number ***n*** is called the size or length of the array.

- The elements of the array are stored respectively in successive memory locations.

Unit-based indexing: The method of numbering the *n*th element of the array with index *n* is called unit-based indexing. In this case the index of each array element is equal to the number of its position in the array.
Zero-based indexing: The method of numbering the *n*th element of the array with index *n-1* is called zero-based indexing. In this case the index of each array element is equal to the number of number of its position in the array minus one.

- This array is called one-dimensional array because each element in such an array is referenced by one subscript. Note that in two-dimensional array, each element is referenced by two subscripts.

Prepared by: K M Akkas Ali, Assistant Professor, IIT, JU

Data Structure: Array

- Three numbers define an array : lower bound, upper bound, size.

1. **Lower bound:**
   - The lower bound is the smallest subscript or index you can use in the array (usually 0).

2. **Upper bound:**
   - The upper bound is the largest subscript or index you can use in the array.

3. **Size:**
   - The size or length of the array refers to the number of data elements in the array. It can be computed from the index set as:

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

Where,

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

- Note that, length = UB, when LB = 1.

Prepared by: K M Akkas Ali, Assistant Professor, IIT, JU

Data Structure: Array

Notations used in Array:

- If STUDENT is the name of an array and there are a total of n data elements, then each element of this array is denoted by one of the following three notations:

1. **Subscript notation:**
   $STUDENT_1$, $STUDENT_2$, $STUDENT_3$, ………$STUDENT_n$ or,
   $STUDENT_0$, $STUDENT_1$, $STUDENT_2$, ………$STUDENT_{n-1}$

2. **Parenthesis notation:**
   STUDENT(1), STUDENT(2), STUDENT(3), ……..STUDENT(n) or,
   STUDENT(0), STUDENT(1), STUDENT(2), ……..STUDENT(n-1)

3. **Bracket notation:**
   STUDENT[1], STUDENT[2], STUDENT[3], ………..STUDENT[n] or,
   STUDENT[0], STUDENT[1], STUDENT[2], ………..STUDENT[3]

- Regardless of the notation,
  - ✓ the number 1 or 2 in STUDENT[1] or STUDENT[2] is called a *subscript* or an *index* that locates the position of the element within the array.
  - ✓ STUDENT[1] or STUDENT[2] is called a *subscripted variable* that represents 1st or 2nd element of the array.
  - ✓ STUDENT is the *name of the array*.

Prepared by: K M Akkas Ali, Assistant Professor, IIT, JU

Data Structure: Array

Example-1:

- Suppose STUDENT is the name of a one-dimensional array that consists of seven students. This array is depicted either horizontal or vertical position shown in the figure below:

- Here, STUDENT[1] denotes Asif Khan, STUDENT[2] denotes Sumon Sarker, STUDENT[3] denotes Nusrat Jahan, and so on.

STUDENT

1	Asif Khan
2	Sumon Sarker
3	Nusrat Jahan
4	Ahmedul Kabir
5	Dhanita Tripura
6	Shakila Zaman
7	Golam Rabbani

STUDENT

Asif Khan	Sumon Sarker	Nusrat Jahan	Ahmedul Kabir	Dhanita Tripura	Shakila Zaman	Golam Rabbani
1	2	3	4	5	6	7

Figure: One-Dimensional Array

Prepared by: K M Akkas Ali, Assistant Professor, IIT, JU

Data Structure: Array

Example-2:

- An automobile company uses an array AUTO to record the number of automobiles sold each year from 1932 through 1984. Rather than beginning index set with 1, it is more useful to begin the index set with 1932 so that

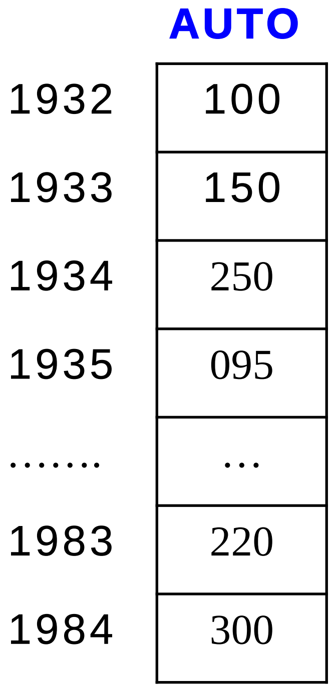
  AUTO[K] = number of automobiles sold in the year K

- Then LB=1932 is the lower bound and UB=1984 is the upper bound of the array AUTO.

- Therefore, the length of the array is:

  Length= UB – LB +1 = 1984-1930 +1 = 55

- That is, the array AUTO contains 55 elements and its index set consists of all integers from 1932 through 1984.



	AUTO
1932	100
1933	150
1934	250
1935	095
…….	…
1983	220
1984	300

Figure: One-Dimensional Array

Data Structure: Array

Multi-dimensional Array:

- It is a collection of similar data elements where each element is referenced by m subscripts. For example, in two-dimensional array, each element is referenced by two subscripts. Such arrays are called matrices in mathematics and tables in business applications.

Example:

- Suppose two tutorial tests were taken for seven students of IIT 4th batch.
- Let the students are numbered from 1 to 7, and class roll, tutorial-1 and tutorial-2 are numbered from 1 to 3.
- Such data can be stored in the computer using a two-dimensional array that is depicted in the figure below.
- If the array name is given CLASTEST, then
  - ❑ CLASSTEST[1,1] denotes the class roll of first student, i.e. 826.
  - ❑ CLASSTEST[1,2] denotes the marks of first student on the 1st tutorial, i.e. 35.
  - ❑ CLASSTEST[1,3] denotes the marks of first student on the 2nd tutorial, i.e. 24.
  - ❑ CLASSTEST[2,1] denotes the class roll of second student, i.e. 827.
  - ❑ CLASSTEST[2,2] denotes the marks of second student on the 1st tutorial, i.e. 26.
  - ❑ CLASSTEST[2,3] denotes the marks of second student on the 2nd tutorial, i.e. 36.

CLASSTEST

	Class Roll	Tutorial-1	Tutorial-2
1	826	35	24
2	827	26	36
3	828	28	30
4	829	33	31
5	830	30	32
6	831	20	28
7	832	34	22

Figure: Two-Dimensional Array

- The size of the above array is denoted by 7x3 (read as 7 by 3), since it contains 7 rows and 3 columns.

Data Structure: Array

Properties of an Array:

- An array has the following properties:
  - An array is a fixed-size data structure.
  - An array is a sequence collection of data elements.
  - It can hold multiple values of the same type.
  - An array must have a name.
  - The name of the array holds the address of the first array element.
  - Elements of an array must be finite.
  - Elements of an array must be of similar type.
  - Each element of an array is referenced by an index.
  - Indexing generally begins at zero.
  - Each element of an array is referenced by notation like A(1), A(2), A(3) etc. or A_1, A_2, A_3, etc. or A[1], A[2], A[3] etc. where A is an array.
  - Elements of the array are stored respectively in successive memory locations.
  - We specify the array size at compile time, often with a named constant.
  - Array can be indexed. (Linked list can not be indexed).
  - It enables us to develop a concise and efficient programs.

Prepared by: K M Akkas Ali, Assistant Professor, IIT, JU

Data Structure: Array

Declaring an Array:

- Each programming language has its own rules for declaring arrays. Each such declaration must give three items of information implicitly or explicitly:
  1. Name of the array
  2. The data type of the array
  3. The index set of the array
- Some programming language (e.g., FORTRAN and Pascal) allocate memory space for arrays statically, i.e., during program compilation; hence the size of the array is fixed during program execution. On the other hand, some other programming languages (e.g., C, C++) allocate memory space for arrays dynamically, i.e., they allow one to read an integer ***n*** and then declare an array with ***n*** elements.

Example:

- In C, arrays must be declared before they are used in the programs. The general form of array declaration in C is:

  type variable-name[***size***];

  where ***type*** indicates the type of element that will be contained in the array, such as ***int***, ***float***, or ***char; size*** represents the maximum number of elements that can be stored inside the array. For example,

  float height[***50***];

  declares the **height** to be an array containing 50 real elements. Any subscripts from 0 to 49 are valid.

Prepared by: K M Akkas Ali, Assistant Professor, IIT, JU

Data Structure: Array

Representation of Linear Arrays in Memory:

- Let AUTO be a linear array in the memory of a computer which is simply a sequence of addressed locations as pictured in figure below.
- Let us use the notation

LOC (AUTO[K]) = address of element AUTO[K] of the array AUTO

- Though the elements of the array AUTO are stored in successive memory cells, the computer does not need to keep track of the address of every element of AUTO, but needs to keep track only of the address of the first element of the array (called the base address of the array AUTO), which is denoted by,

Base(AUTO)

- The computer calculates the address of any element of the array using this base address by the following formula:

LOC(AUTO[K]) = Base(AUTO) + w(K- LB)

Where w is the number of words per memory cell for the array AUTO.

- Note that, the time to calculate LOC(AUTO[K]) is essentially the same for any value of K.
- If any value of the subscript K is given, one can locate and access the content of AUTO[K] without scanning any other element of the array.

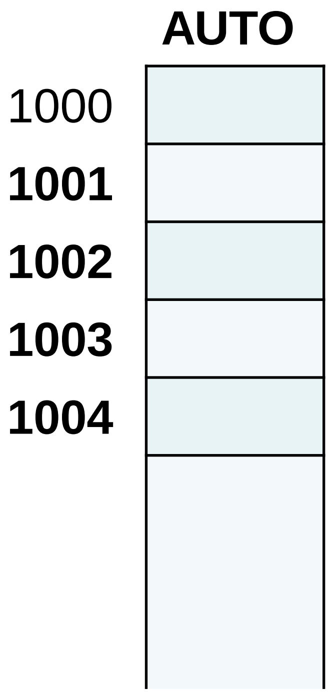


Figure: Computer Memory

Data Structure: Array

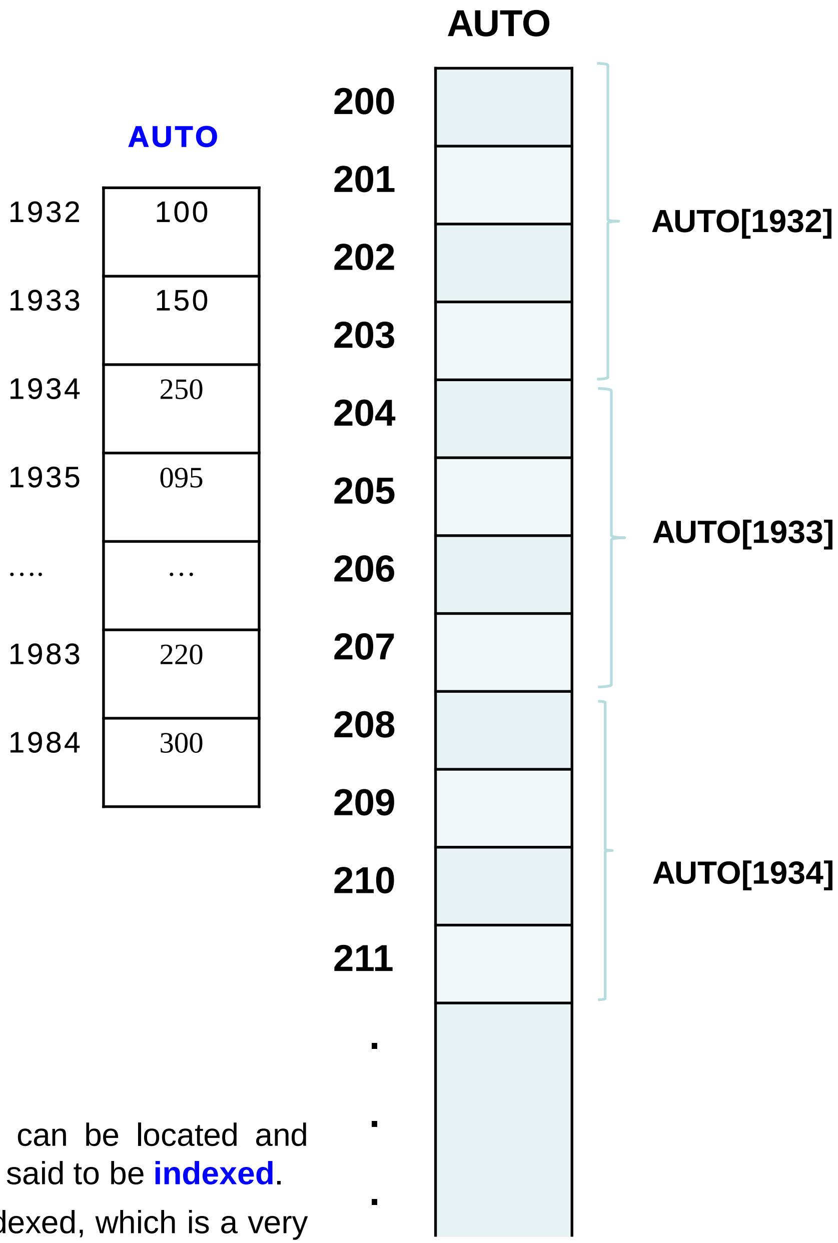
Representation of Linear Arrays in Memory:

Example:

- ➢ An automobile company uses an array AUTO to record the number of automobiles sold each year from 1932 through 1984. Suppose AUTO appears in memory as pictured in figure below.
- ➢ Here, Base(AUTO) = 200, and w = 4 words per memory cell for AUTO.
- ➢ Therefore,

  LOC(AUTO[1932]) = 200, LOC(AUTO[1933]) = 204, LOC(AUTO[1934]) = 208, ….
- ➢ In similar ways, the address of the array element for year K = 1965 can be obtained by the following formula:

  LOC(AUTO[1965]) = Base(AUTO) + w(1965 – LB) =200 + 4(1965 – 1932) = 332
- ➢ Therefore, it is obvious that the contents of this element can be obtained without scanning any other element in the array AUTO.



	AUTO
1932	100
1933	150
1934	250
1935	095
….	…
1983	220
1984	300

Figure: Memory representation of an array

Note:

- ➢ If any element of an array (e.g. AUTO[K] of the array AUTO) can be located and processed in a time that is independent of K, then the array is said to be **indexed**.
- ➢ The above example demonstrates that linear array can be indexed, which is a very important property of linear array. In fact, linked list can not be indexed.

Have a question?

Thank you...

Prepared by: K M Akkas Ali, Assistant Professor, IIT, JU