CC4

Chapter II: Data Structures and Address Calculations

Before we start

- **Data Types**
  - Refers to different kinds of data that a variable may hold in a Programming Language
    - Examples:

Data Type	Memory Allocation
Char	1 byte
Int	2 bytes
Float	4 bytes
Double	8 bytes

Data Structures

- Are ways of **storing** and **organizing data** for **efficient utilization**
- Works **together** with **algorithms** to **reduce time and space complexity** while solving **problems**
- Describe how a set of objects are **related** and the set of **operations** that can be applied on the elements

- **Array**
- **Linked-List**
- **Record**
- **Matrix**
- Stack
- Queue
- Binary Tree
- Binary Search Tree
- AVL Tree
- Graph

College of Information Technology and Computer Science
CENTER OF EXCELLENCE in Information Technology

by: RBMB,FAP

3

Arrays

- Stores **homogenous data** at **contiguous memory addresses** or locations
- Memory space allocated according to **initialized array size**
- Can be of **n-dimensions,** where n is any number from 1 onwards
- Although hidden from us, since it is contiguous, **address calculations** of elements can be determined (later)

College of Information Technology and Computer Science
CENTER OF EXCELLENCE in Information Technology

by: RBMB,FAP

4

Arrays

- Memory space is not allocated upon declaration, instead upon construction when size is known during instantiation using the keyword *new* in Java.

```
Int [] hardy;
hardy=new int[5];
```

In single line

```
Int[] hardy=new int[5];
```

Arrays

- One-dimensional Array
- Two-dimensional Arrays
- Multi-dimensional Arrays

Arrays

- Array Address Calculation Formulae (For Lab Activity#2)

To determine the ith element of a Single-dimension array:

$$A[i] = \alpha + (i) * esize$$

where:

α - base or starting address
i - element
esize - element size in bytes

Example: Determine the address of 5th element of an integer array A with a starting address of 2000

7

Arrays

- Array Address Calculation Formulae (For Lab Activity#2)

To determine the ith element of a Two-dimension array:

$$A[i][j] = \alpha + [(i)*(UB_2)+(j)] * esize$$

where:

UB_2 - upper bound of the 2nd dimension
α - base or starting address
esize - element size in bytes

8

Arrays

- Array Address Calculation Formulae (For Lab Activity#2)

To determine the ith element of a Three-dimension array:

$$A[i][j][k] = \alpha + [(i)*(UB_2)*(UB_3)+(j)*(UB_3)+(k)]*esize$$

where:

UB_3 – upper bound of the 3^{rd} dimension
UB_2 – upper bound of the 2^{nd} dimension
α - base or starting address
esize – element size in bytes

College of Information Technology and Computer Science
CENTER OF EXCELLENCE in Information Technology

by: RBMB,FAP

9

Arrays

- Exercise:

1. Given A[10][3][3][6], α =2000, esize=4 bytes:
   a. find the formula to represent an element in a 4-dimensional array.
   b. find the total number of elements
   c. find the address of A[2][2][0][4]

2. Given X[8][3][6][2][3], α =3000, esize=3 bytes:
   a. find the total number of elements
   b. find the address of X[0][2][5][1][2]

College of Information Technology and Computer Science
CENTER OF EXCELLENCE in Information Technology

by: RBMB,FAP

10

Arrays

- Problem:

Snow Enterprises gives a fixed salary of P1500 per week to all their newly hired salespeople. They also give an additional 8.5% bonus for each salesperson's Gross Weekly Sales. The company classifies these new employees depending on their Net Weekly Salary as follows:

Employee Classification	Net Weekly Salary
A	P1500 – P1999
B	P2000 – P2499
C	P2500 – P2999
D	P3000 – P3499
E	P3500 – P3999
F	P4000 and above

Write a program that accepts a user's Name and Gross Weekly Sales as inputs. Your program should only have 2 methods outside of the main method:

a. Create a *netweekly* method that accepts Gross Weekly Sales as a parameter and returns the Net Weekly Salary as stated in the problem definition.
b. Create a *classify* method that accepts Net Weekly Salary as a parameter and returns the Employee Classification as per the table in the problem definition.

Optional: If your *classify* method can perform classification using arrays, only 1 loop and 1 condition, additional points will be credited to your next activity.

College of Information Technology and Computer Science
CENTER OF EXCELLENCE in Information Technology

by: RBMB,FAP

11

Linked Lists

- Consists of 2 values:
  - **Node**: holds **actual element value**
  - **Link**: holds a **reference** to the **next node,** if reference is NULL, end of list
- Memory addresses are allocated **as needed** when **nodes are added**
- Can be **Singly Linked List**, **Doubly Linked List**, **Circular Linked List**

College of Information Technology and Computer Science
CENTER OF EXCELLENCE in Information Technology

by: RBMB,FAP

12

Records

- Stores **heterogenous data**
- Each of these datum is referred to as **fields**
- A combination of records usually create a **database**

13

Records

Declaration:

```
typedef struct
{
     <data type1> field1;
     <data type2> field2;
     <data type3> field3;
     <data typeN> fieldN;
}RecordType;
```

RecordType

field1
field2
field3
.
.
fieldN

14

Matrices

- A **matrix** is a rectangular or square grid of numbers arranged into rows and columns.

- Can be generalized to be 2D-arrays you can perform arithmetic calculations on: Addition, Subtraction, Multiplication, and Transposition.

Matrix M $\quad M = \begin{bmatrix} 4 & 1 & 5 \\ 3 & 6 & 2 \end{bmatrix}$

Matrix Operations

Matrix Addition

- **RULES:**
  1. Matrices must have the **same dimensions**
  2. Add the **values** of each element at the **same position**

$$\begin{bmatrix} \mathbf{8} & 4 & 2 \\ 6 & 1 & 5 \end{bmatrix} + \begin{bmatrix} \mathbf{3} & 10 & 4 \\ 5 & 6 & 1 \end{bmatrix} = \begin{bmatrix} \mathbf{11} & 14 & 6 \\ 11 & 7 & 6 \end{bmatrix}$$

Matrix Operations

Matrix Subtraction

- **RULES:**
  1. Matrices must have the **same dimensions**
  2. Subtract the **values** of each element at the **same position**

$$\begin{bmatrix} \mathbf{8} & 4 & 2 \\ 6 & 1 & 5 \end{bmatrix} - \begin{bmatrix} \mathbf{3} & 10 & 4 \\ 5 & 6 & 1 \end{bmatrix} = \begin{bmatrix} \mathbf{5} & -6 & -2 \\ 1 & -5 & 4 \end{bmatrix}$$

College of Information Technology and Computer Science
CENTER OF EXCELLENCE in Information Technology

by: RBMB,FAP

17

Matrix Operations

Matrix Multiplication (with a Constant)

- **RULES:**
  1. Multiply each element with the constant value

$$2 \times \begin{bmatrix} 4 & 1 \\ 3 & 6 \end{bmatrix} = \begin{bmatrix} 8 & 2 \\ 6 & 12 \end{bmatrix}$$

$$-1 \times \begin{bmatrix} -4 & 1 \\ -3 & 5 \\ 6 & -2 \end{bmatrix} = \begin{bmatrix} 4 & -1 \\ 3 & -5 \\ -6 & 2 \end{bmatrix}$$

College of Information Technology and Computer Science
CENTER OF EXCELLENCE in Information Technology

by: RBMB,FAP

18

Matrix Operations

Matrix Multiplication

- **RULES:**
  1. Matrices can **only** be multiplied **if** the **# of columns** in the **first matrix** is **equal** to the **# of rows** in the **second matrix**.

$$A = \begin{bmatrix} 4 & -1 & 5 \\ 3 & 6 & -2 \end{bmatrix} \qquad B = \begin{bmatrix} 3 \\ 6 \\ -2 \end{bmatrix}$$

2 x **3**   **3** x 1

College of Information Technology and Computer Science
CENTER OF EXCELLENCE in Information Technology

by: RBMB,FAP

19

Matrix Operations

Matrix Multiplication

- Therefore, multiplying matrices is **not commutative**

$$B = \begin{bmatrix} 3 \\ 6 \\ -2 \end{bmatrix} \qquad A = \begin{bmatrix} 4 & -1 & 5 \\ 3 & 6 & -2 \end{bmatrix}$$

3 x **1**   **2** x 3

College of Information Technology and Computer Science
CENTER OF EXCELLENCE in Information Technology

by: RBMB,FAP

20

Matrix Operations

Matrix Multiplication

- **RULES:**

  2. **Sum of the products** of elements are computed from **rows** of **first matrix** and **columns** of **second matrix**.

Let matrix $A = \begin{bmatrix} a & b \\ c & d \end{bmatrix}$ and $B = \begin{bmatrix} e & f \\ g & h \end{bmatrix}$

$$A(B) = \begin{bmatrix} \mathbf{a} & \mathbf{b} \\ c & d \end{bmatrix} \begin{bmatrix} \mathbf{e} & f \\ \mathbf{g} & h \end{bmatrix} = \begin{bmatrix} \mathbf{ae + bg} & af + bh \\ ce + dg & cf + dh \end{bmatrix}$$

Matrix Operations

Matrix Multiplication

$A = \begin{bmatrix} 5 & 1 & 6 \\ 0 & 8 & -2 \end{bmatrix}$ $\quad B = \begin{bmatrix} 2 \\ 3 \\ 4 \end{bmatrix}$

2 x **3** $\quad$ **3** x 1

$$A(B) = \begin{bmatrix} 5(2) + 1(3) + 6(4) \\ 0(2) + 8(3) + (-2)(4) \end{bmatrix} = \begin{bmatrix} 37 \\ 16 \end{bmatrix}$$

- Notice A(B) is now a 2 x 1 matrix.

Matrix Operations

Matrix Transposition

- Swap the rows for the columns

$$\begin{bmatrix} 4 & 1 & 5 \\ 3 & 6 & 2 \end{bmatrix}^T = \begin{bmatrix} 4 & 3 \\ 1 & 6 \\ 5 & 2 \end{bmatrix}$$

Matrix Operations

Exercises

Given:

$$A = \begin{bmatrix} 3 & 1 & 5 \\ 6 & 2 & 0 \end{bmatrix} \quad B = \begin{bmatrix} 6 \\ 4 \\ -1 \end{bmatrix} \quad C = \begin{bmatrix} 2 & 4 \\ 3 & 6 \\ -1 & 2 \end{bmatrix} \quad D = \begin{bmatrix} 5 & 2 \\ 3 & 1 \end{bmatrix} \quad E = \begin{bmatrix} 3 & -2 \\ 1 & 4 \end{bmatrix}$$

$$F = \begin{bmatrix} 2 & 1 & 3 \\ 5 & 7 & -2 \end{bmatrix}$$

1. A + F
2. E – D
3. C + B

4. C(D)
5. A(F)
6. C^T

7. F^T(E)