Attendance record

- To check what devices students are using to join class
- To check attendance

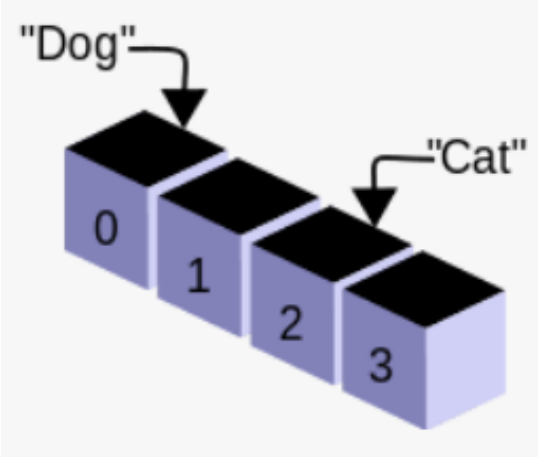
https://forms.gle/uwkS432JgpowVSBx5

Start class: **3:05pm**


QUIZ!

ALGORITHM & PROGRAMMING

Chapter 4- Array



Prepared by: Dr. VALY Dona and Mr. NOU Sotheany

Modified and Lectured by: Mr. BOU Channa

Lecture overview

❑ Overall lectures

Mid-term

1. Introduction to algorithm
2. Basic data types and statements
3. Control structures and Loop

Final exam

4. **Array**
5. Data structure
6. Sub-programs

Semester I

Lesson objectives

❑ Objectives

- Upon completion of this lesson, students will be able to
  - Understand a more advance type of data, called **Array**
    - Array of number
    - Array of string
    - Array of character …
  - Know how to use different kind of array

Outline

- An overview of the lessons

- Introduction
  - Problem when not using array
- Array
  - What is array?
  - How to use array?
  - More on array

Introduction

❑ Problem

- **Problem #1**: Suppose we want to get 100 students' names then display theirs names in a list.

Will you use 100 variables?

```
Var name1, name2, ..., name100 : Sequence of characters
Begin
   read(name1, name2, ..., name100)
End
```

Disadvantages:
Too many creation of variables?
What if we have more than 100 variables?

- **Problem #2**: Suppose we want to get 100 subjects' scores of a students then do summation of those score.

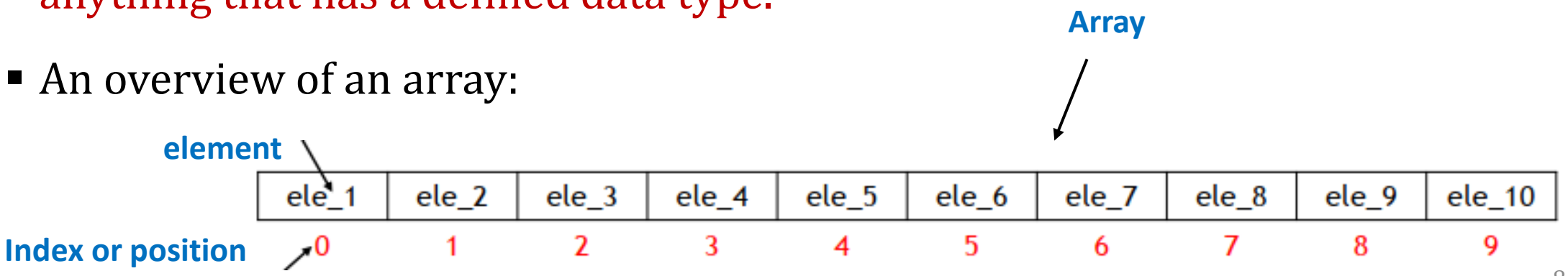
Will we need 100 variables to store those scores?

```
Var score1, score2, …, score100, sum : float
Begin
   read(score1, score2, ..., score100)
   sum ← score 1+ score2 + … + score100
End
```

Array

❑ What is an array?

- Array is a kind of data structure that stores many variables (elements) as a **single special variable.**
- **Each variable** in an array is called an **array element** and they have the **same variable type**
- You could have an **array of integers** or an **array of characters** or an array of anything that has a defined data type.
- An overview of an array:



Array

❑ Declaring (creating) an array

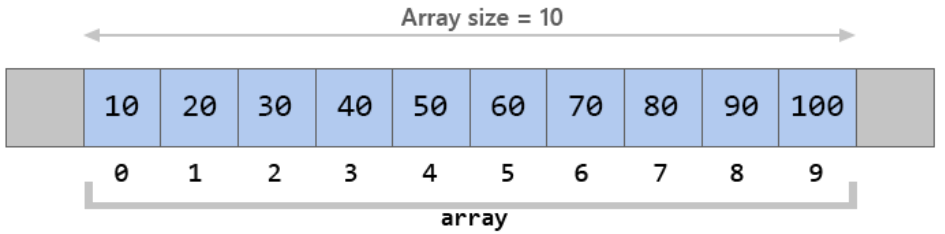
- To declare an array, we have to choose
  - Type of element in the array
  - Number of elements in the array
- Syntax

```
Var identifier[number of elements] : Type of element in array
```

- Examples: Creating array

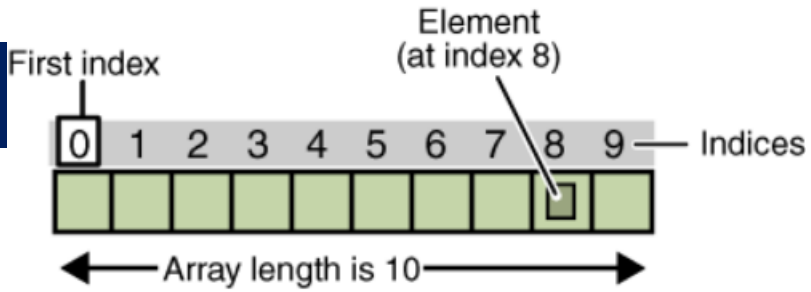
```
Var num[20] : Integer
Var scores[10] : Float
Var name[50] : Array/sequence of characters
Var s[5][100] : Array of string (5 elements)
```

Suppose that we add values (10, 20, .., 100) to the array.
The array now look like this:



Array

❑ Index or position



- In array, the value of index is
  - Start from 0 (some language may start with index 1)
    - **E.g**: In C language, index starts from 0 **but** in Matlab index starts from 1
  - Integer number
  - Last value of index is equal to number of elements in array minus 1

  (when its index starts with 0)
- Index in the bracket can either be a direct integer value or a variable or an expression

```
Var score[10] : Integer
Begin

     score[0] ← 70
     score[1] ← 80
End
```

```
Var n : Integer
Var score[10] : Integer
Begin
     n ← 0
     score[n]   ← 70
     score[n+1] ← 80
End
```

Array

❑ Access/use to an array

- To display array's elements, we need to access to each element
- To access a specific element in an array, use **arrayName[index]**
  - Ex: Suppose the array named **ele**
  - Then to access: **ele**[0], **ele**[1], …., **ele**[9]
- Examples

```
Var scores[10] : Float
Begin
   read(scores[0])
   read(scores[1])
   write("Score student 1: ", scores[0])
   write("Score student 2: ", scores[1])
End
```

```
Var i : Integer
Var num[10] : Integer
Begin
     for (i←0; i<10; i←i+1) do
        read(num[i])
     end for

     for (i←0; i<10; i←i+1) do
        write(num[i])
     end for
End
```

What does this algorithm do?

Array

❑ Access/use to an array

- To access a specific element in an array, use **arrayName[index]**
  - Suppose we have an array named **ele**
  - Usage: **ele**[0], **ele**[1], …., **ele**[9]

What does these algorithms do?

- Get 10 numbers from the user.
- Then sum all those numbers together.
- Finally, display the result.

```
Var i : Integer
Var num[10] : Integer
Var s: Integer
Begin
     for (i←0; i<10; i←i+1) do
        read(num[i])
     end for
     s←0
     for (i←0; i<10; i←i+1) do
         s ← s + num[i]
     end for
     write(s)
End
```

- Get 10 gender from the user.
- Then count all males and females
- Finally, display display #male, #female

```
Var i : Integer
Var gender[10] : Sequence of character
Var m, n: Integer
Begin
     for (i←0; i<10; i←i+1) do
        read(gender[i])
     end for
     m←0
     n←0
     for (i←0; i<10; i←i+1) do
        if gender[i]==‘M’ then
             m++
        else if (gender[i]==‘F’ then
             n++
        end if
     end for
     write(m, n)
End
```

Array

❑ Using array to solve the previous problems?

- **Solution for Problem #1**:
  - *Use an array with the size of 100 and its type is a string (sequence of characters)*

```
Var names[100][20] : Sequence of characters
Begin
   for(i←0; i<=99; i++) do
       read(names[i])
   end for
End
```

- **Solution for Problem #2**:
  - *Use an array with the size of 100 and its type is a float*
  - Combine those variables into one by declaring an array then do loop to find summation.

```
Var scores[100] : float
Var sum : float
Begin
   sum ← 0
   for(i←0; i<=99; i++) do
       read(scores[i])
   end for

   for(i←0; i<=99; i++) do
       sum ← sum + scores[i]
   end for
   write("Total scores: ", sum)
End
```

Example 1

ection B.c | Array.c | Ex5 correction B.c | While loop.c | while loop A.c | Ex5 correction C.c | Number prediction prog

```
1
2   #include<stdio.h>
3
4   main(){
5       int n[7];
6
7       //Get input numbers and store in array
8       for(int k=0; k<=6; k=k+1){
9           printf("Enter number #%d: ",k+1);
10          scanf("%d", &n[k]);
11      }
12
13      //Display data in array
14      printf("\n\n");
15      for(int p=0; p<=6; p=p+1){
16          printf("%d ", n[p]);
17      }
18
19
20  }
21
```

Select C:\!Data\Datastructure\Array.exe

```
Enter number #1: 9
Enter number #2: 2
Enter number #3: 5
Enter number #4: 0
Enter number #5: -12
Enter number #6: 98
Enter number #7: 100


9 2 5 0 -12 98 100
Process returned 0 (0x0)
Press any key to continue.
```

Practice

❑ Practice exercises

1. Write algorithms for the problems below :
   a. Declare and store an array with 5 English's vowels
   b. Declare and store an array with English's alphabet A-Z
   c. Declare and store an array with even integer numbers 2, 4, … 100
   d. Declare and store an array of 10 user names. Ask the user to input all those 10 names. Then display their names on the screen

2. Write an algorithm to ask a user for 20 scores then
   - Find the average of those scores and show the scores that are greater than the average

Practice -HOMEWORK

Write a C program to ask a user for 20 scores then

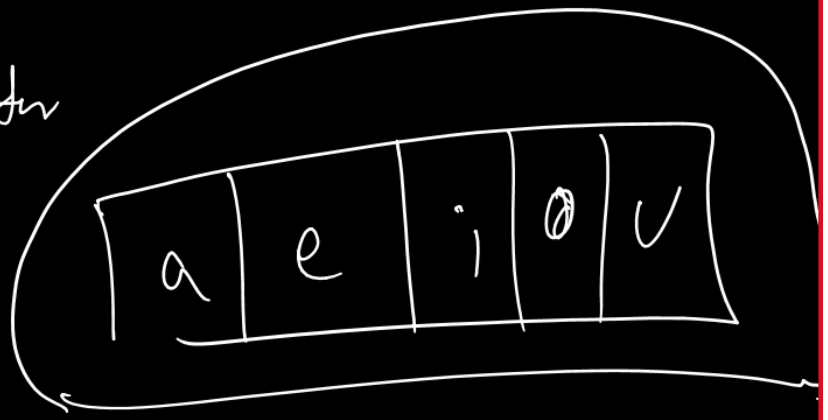
- Find the **average** of those scores
- Show the scores that are greater than the average
- Count number of students who got score more than average

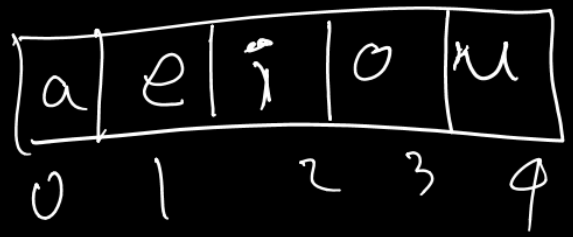
```
Input number #1: 10
Input number #2: 20
.....
Input number #20: 200

=>OUTPUT: Average is: 105. Scores that are more than average are: 110, 120,
130,140,150,160,170,180,190,200
```

Coding hour
HOMEWORK
Deadline: Tomorrow night

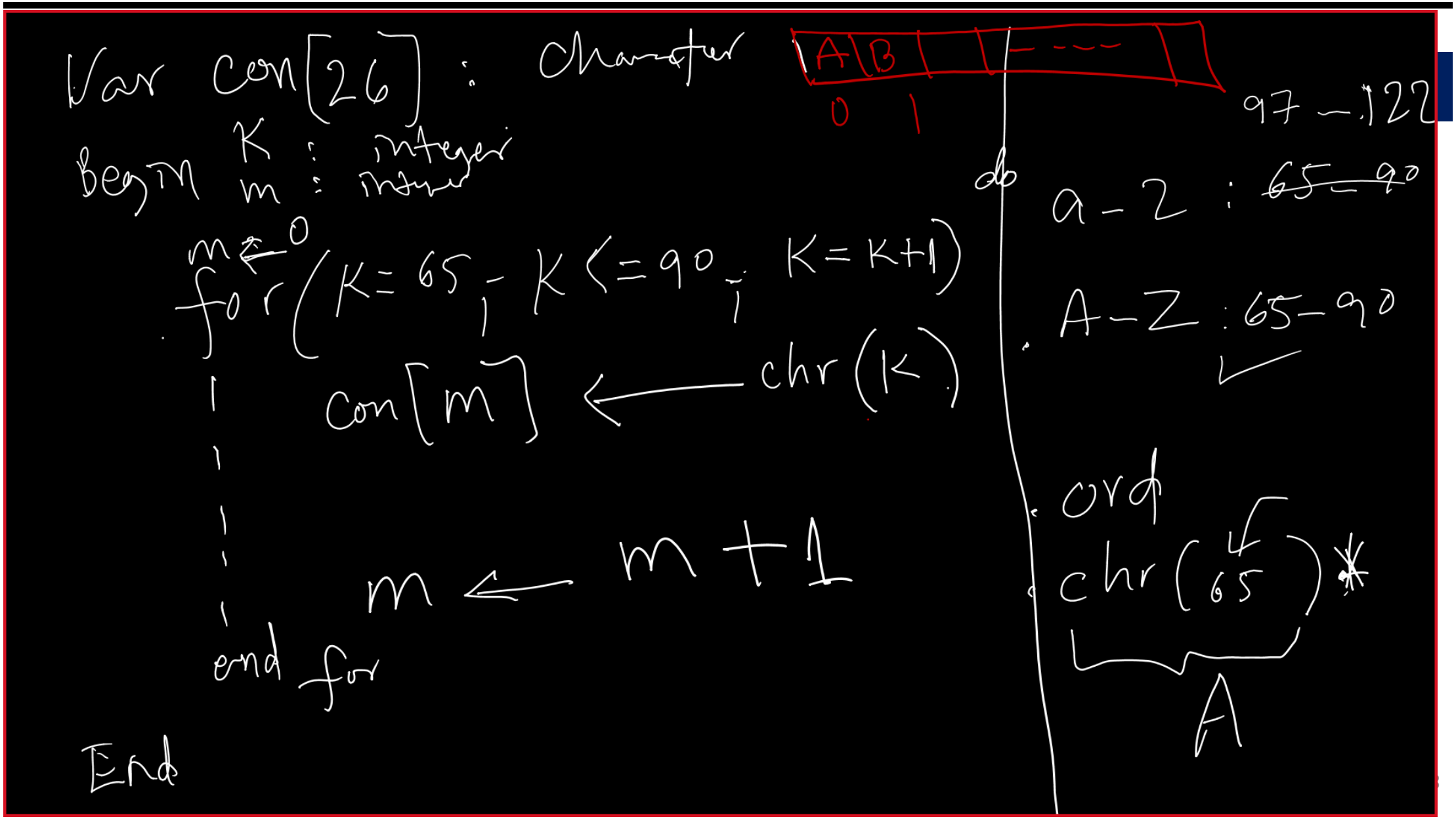
Practice

```
Var Vowel[5] : ~~Sequence of~~ Character
Begin
    vowel[0] ← 'a'
    vowel[1] ← 'e'
    vowel[2] ← 'i'
    vowel[3] ← 'o'
    vowel[4] ← 'u'
End
```



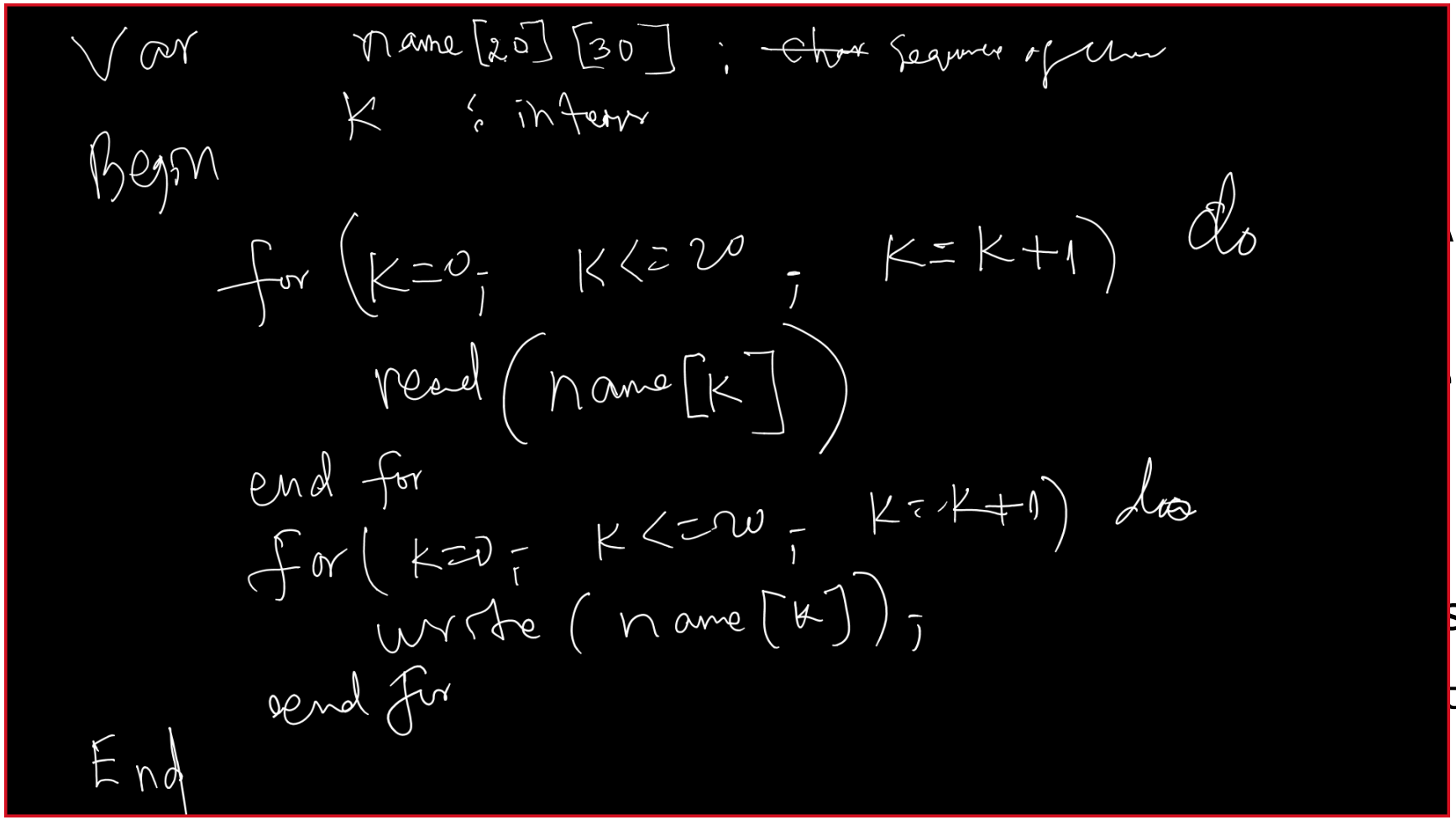


Practice


Var con[26] : character
Begin K : integer
m : integer
m ← 0
for (K = 65 ; K <= 90 ; K = K+1)
con[m] ← chr(K)
m ← m + 1
end for
End
A B
0 1
db
a-z : 65-90
A-Z : 65-90
ord
chr(65) *
A
97 - 122

Practice

❑ Practice exercises



-Z

2, 4, … 100

user to input

screen

s then

that are greater

TP -Array

Deadline: ?

❑ Practice exercises

1. Write C programs for the problems below :
   a. Declare and store an array with 5 English's vowels
   b. Declare and store an array with English's alphabet A-Z
   c. Declare and store an array with even integer numbers 2, 4, … 100
   d. Declare and store an array of 10 user names. Ask the user to input all those 10 names. Then display their names on the screen

2. Write a C program to ask a user for 20 scores then
   - Find the average of those scores and show the scores that are greater than the average

Practice

❑ Array

1. Write an algorithm to get 10 input numbers and store in an array using a for loop.

2. With the extension to exercise #1, find the largest and second largest numbers in the array and display on screen.

Practice

❑ Array

3. Write an algorithm to store the word “**New York City**” in an array of characters. Then make it to lowercase and store in another array.

Remark:

- Do not use the function **tolowercase**
- Use operation with ASCII code

```
int  n[10];
int p[7][10];
```

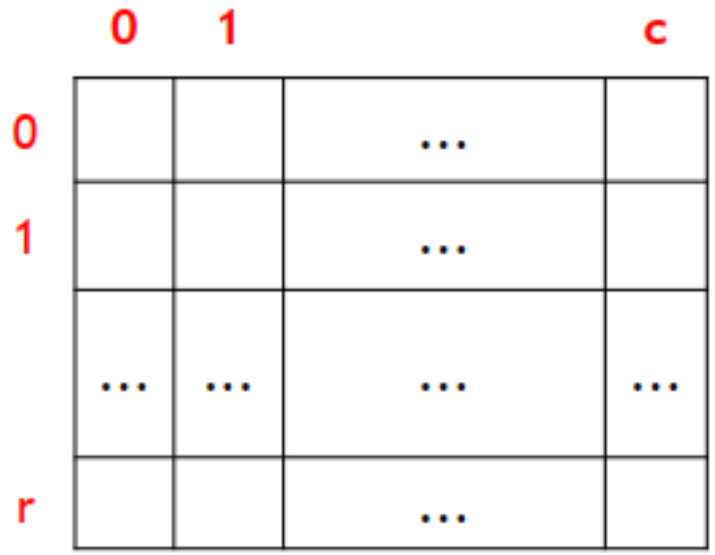
Two-dimensional array

Break 15mn
Start 9am

Two-dimensional array

❑ What?

- Two-dimensional array is an array which is represented as same as a table. It is composed of #columns ***c*** and #rows ***r*** .
- A two-dimensional array consists of ***r*** * c elements



	Column 1	Column 2	Column 3	Column 4
Row 1	x[0][0]	x[0][1]	x[0][2]	x[0][3]
Row 2	x[1][0]	x[1][1]	x[1][2]	x[1][3]
Row 3	x[2][0]	x[2][1]	x[2][2]	x[2][3]

Two-dimensional array

❑ What?

- Declaration

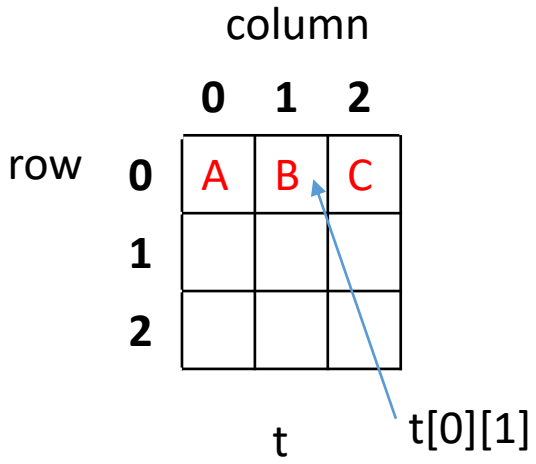
#row

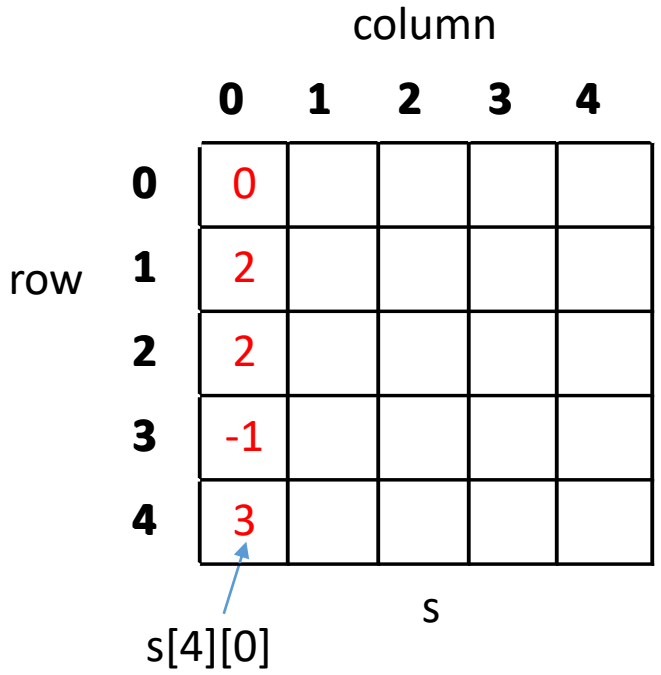
#column

```
Var t[3][3] : characters
Var s[5][5] : Integer
```

- Usage

```
t[0][0] <- 'A'
t[0][1] <- 'B'
t[0][2] <- 'C'

s[0][0] <- 1
s[1][0] <- 2
s[2][0] <- s[0][0]+1
s[3][0] <- 1-s[1][0]
s[4][0] <- 3
s[0][0] <- s[0][0]-1
```





Example

- Store values of a matric

```
Var i, j, m1[3][3] : integer
Begin
   for (i←0; i<3; i++) do
      for (j←0; j<3; j++) do
        read(m1[i][j])
      end for
   end for

End
```

Store values from input in 2-dimentional array

```
for (i←0; i<3; i++) do
   for (j←0; j<3; j++) do
     write(m1[i][j], “ ”)
   end for
end for
```

Display values in 2-dimentional array

Example

❑ Calculate summation of two matrix 3x3

```
Var i, j, m1[3][3], m2[3][3], sum[3][3] : integer
Begin
    for (i←0; i<3; i++) do
        for (j←0; j<3; j++) do
            read(m1[i][j], m2[i][j])
        end for
    end for

    for (i←0; i<3; i++) do
        for (j←0; j<3; j++) do
            sum[i][j] ← m1[i][j] + m2[i][j]
        end for
        write("\n")
    end for
End
```

Multi-dimensional array

❑ What?

- Multi-dimensional array is a list of one dimensional array
- To make a multi-dimensional array, we need to choose
  - *Type of array's elements*
  - *Number of elements in each dimension*
- Syntax

```
Var identifier[r][c][n]...[m] : Type of element in array
```

Practice: Two dimensional array

1. Store values of multiplication table in the two dimensional array.

Columns

Rows

×	0	1	2	3	4	5	6	7	8	**9**	10
0	0	0	0	0	0	0	0	0	0	0	0
1	0	1	2	3	4	5	6	7	8	9	10
2	0	2	4	6	8	10	12	14	16	18	20
3	0	3	6	9	12	15	18	21	24	27	30
4	0	4	8	12	16	20	24	28	32	36	40
5	0	5	10	15	20	25	30	35	40	45	50
6	0	6	12	18	24	30	36	42	48	54	60
7	0	7	14	21	28	35	42	49	56	**63**	70
8	0	8	16	24	32	40	48	56	64	72	80
9	0	9	18	27	36	45	54	63	72	81	90
10	0	10	20	30	40	50	60	70	80	90	100

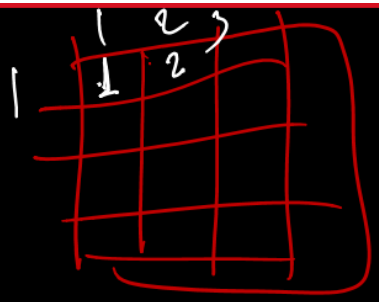
7 x 9 = 63

Var t[10][10] : integer

Begin row, col : integer

for (row = 0; row < 10; row = row + 1) do
  for (col = 0; col < 10; col = col + 1) do
    t[row][col] ← row * col
  end for
end for

End



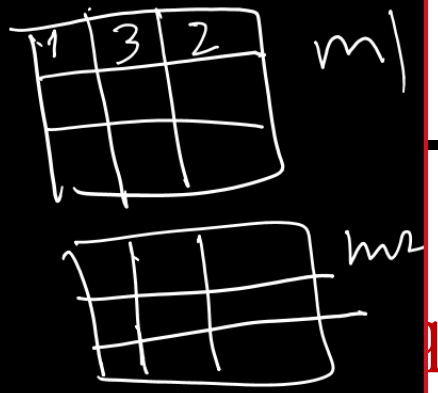
Practice

2. Write an algorithm to check whether two 3x3 matrices are equal. The elements of matrices are given by user.

$$\begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{bmatrix} = \begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{bmatrix}$$

Matrix A Matrix B

```
Var m1[3][3], m2[3][3] : integer
    row, col : integer
Begin
  //cout input
  for (row=0; row<3; row=row+1) do
     for (col=0; col<3; col=col+1) do
        read(m1[row][col], m2[row][col])
     end for
  end for
  //checking
End
```



qual.

Practice

```
// Check if equal
var state := 0 : integer
for (row := 0; row < 3; row := row + 1) do
    for (col := 0; col < 3; col := col + 1) do
        if (m1[row][col] != m2[row][col]) then
            state ← 1
        end if
    end for
end for
if (state == 0) then
    write("Both m1 & m2 are equal matrix")
```

Practice

❑ Practice exercises on Two dimensional array

1. Write an algorithm to check whether a 3x3 matrix is identity matrix
2. Write an algorithm to perform operations on two matrices. For each matrix, ask user for number of rows and number of column. Then ask for the value of each element in each matrix. Do each operation below
   a. Addition
   b. Subtraction
   c. Multiplication

Q & A

Practice

❑ Practice exercises on Two dimensional array

1-D Array

Store in student data in array.

Practice

❑ Practice exercises on Two dimensional array

1. Write an algorithm to check whether two 3x3 matrices are equal. The elements of matrices are given by user.
2. Write an algorithm to check whether a 3x3 matrix is identity matrix
3. Write an algorithm to perform operations on two matrices. For each matrix, ask user for number of rows and number of column. Then ask for the value of each element in each matrix. Do each operation below
   a. Addition
   b. Subtraction
   c. Multiplication