# Data Structures and Object Oriented Programming

## Lecture 1

Dr. Naveed Anwar Bhatti

**Webpage:** naveedanwarbhatti.github.io

# Who am I? Dr. Naveed Anwar Bhatti

**Hometown:** Islamabad

**Last Job:**
Senior Researcher
RISE, Stockholm, Sweden
Joined on April, 2018
**ERCIM Post-Doc (April, 2018 – Sep, 2019)**

**Education:**

**PhD** 2018
Computer Science
Politecnico di Milano, Italy
*System Support for Transiently Powered Embedded Systems*

**MS** 2013
Computer Science
FAST-NUCES, Islamabad, Pakistan
*Long range RFID System: Decoupling sensing and energy in sensor networks using energy transferenc*

**BS** 2011
Telecom
FAST-NUCES, Islamabad, Pakistan
*Internet Controlled Unmanned Ground Vehicle*

# How to reach me?

**Email:** naveed.bhatti@ri.se
**Webpage:** naveedanwarbhatti.github.io
**Class page and slides:** TBD

# Course Objectives: Why are you here?

- Pointers and Arrays

  Week 1

  - Pointers
  - Multi-dimensional Arrays
  - Typecasting (Type conversion)
  - Dynamic Memory Allocation

- User-defined data types

  Week 2

  - User-defined Data Types
  - Typedef
  - Structures
  - Unions

- Object-Oriented Programming in C++

  Week 3, 4 and 5

  - Classes and Objects
  - Classes vs Structure
  - Inheritance
  - Operator Overloading
  - Function and Class templates
  - Method Overriding
  - Virtual Functions

# Course Objectives: Why are you here?

- Introduction of Advanced Data Structures

  Week 6, 7 and 8

  - Array Lists
  - Linked Lists
  - Stacks
  - Queues
  - Trees
  - Vectors (Optional)

- Algorithms

  Week 9, 10, 11 and 12

  - Recursion
  - Searching
    - ❑ Linear
    - ❑ Binary
    - ❑ Depth-First
    - ❑ Breadth-First
  - Sorting
  - Insertion
  - Selection
  - Bubble
  - Merge
  - Heap

# Administrivia and more

- **Pre-requisite**
  - Willingness to work hard!
  - Computer Programming
  - Initiative (very little spoon feeding)

- **Tough Course with fair marking**

# Administrivia and more

- **Grading split**

  - Assignments & class participation 10+5%
  - Quizzes: 5% (start of class, <4 min long)
  - Always bring paper and pen to class
  - First & Second Sessional Exam: 30% each
  - Project: 10%
  - Final Exam: 40%

# Academic integrity

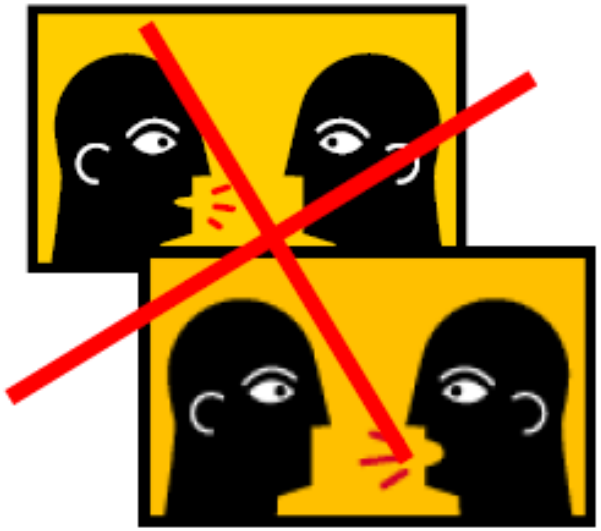
- **Vital to building trust!**
  - Both in you and the university
- **Very serious consequences**
  - In assignment/project will result in a **direct F grade**
  - Code will be checked for similarity
- **A serious offence**
  - Offensive on both religious and secular levels

**Dr. Arshad Ali**
Director HEC

**Dr. Haroon Rashid**
Rector COMSATS

**Dr. Mukhtar Ahmed**
Chairman HEC

# Prohibitions



# Class Attendance and Late arrival

- **University and HEC cares about it**
  - **I do not !**
  - I shall say you are present as long as you tell me before class
  - If you are not serious about the course, its your loss
    - Both money wise
    - And grade wise (directly: 10% participation, quizes indirectly: exams)

- **If you arrive late**
  - Be discrete (come in with minimal fanfare)
  - Be courteous (to other students trying to listen)

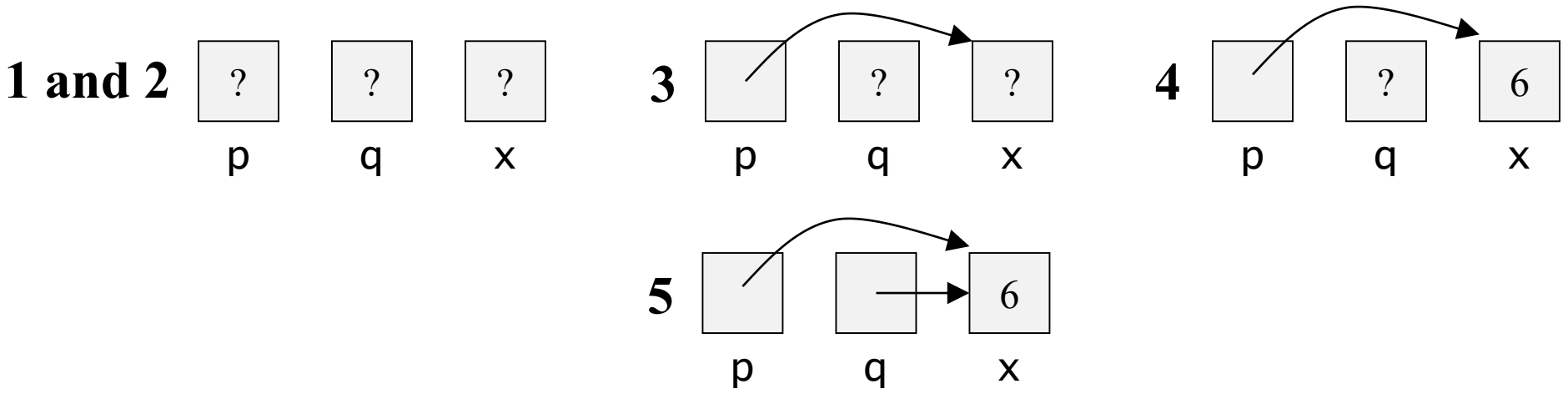
# Late submission policy

- **One Assignment can be submitted late**
  - Total of 72 hrs late submission allowed
  - Any thing greater means your assignment is rejected
  - Choose your late submission carefully.

# Pointers and Arrays

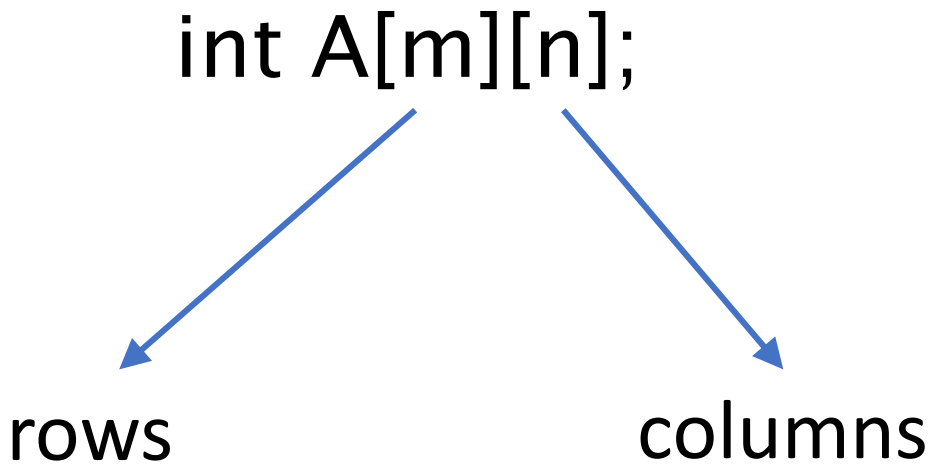
# Pointers (Recap)

| | | |
|---|---|---|
| 1. | Pointer variables | int *p, *q; |
| 2. | Static allocation | int x; |
| 3. | Address-of operator | p = &x; |
| 4. | Memory cell to which P points | *p = 6; |
| 5. | Pointer operations | q = p; |



# Multi-dimensional Arrays

- An array is a contiguous block of memory.
- A 2D array of size m by n is defined as:




What is the number of bytes necessary to hold **int A[2][2]** ?

# Multi-dimensional Arrays

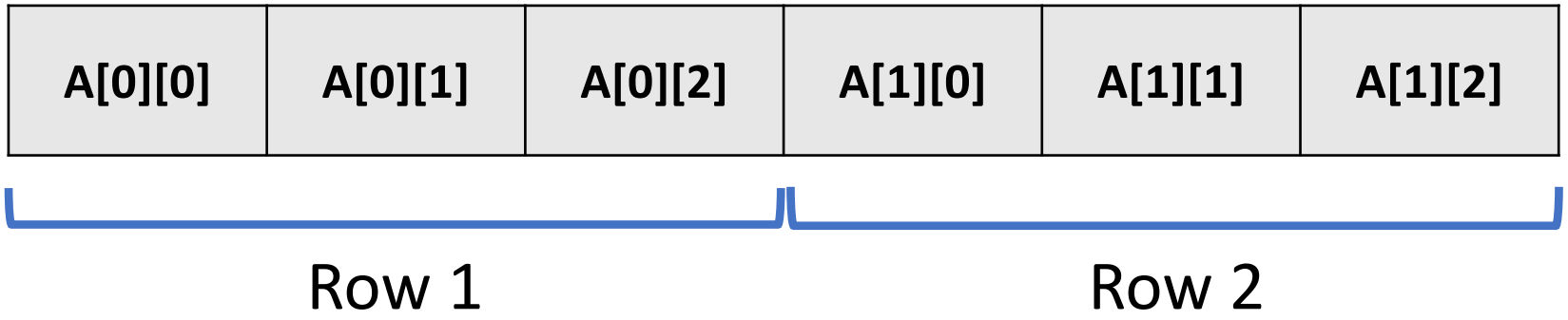
- It can be initialized as:

$$A[2][3]=\{\{1,2,3\},\{4,5,6\}\};$$

| | [0] | [1] | [2] |
|---|---|---|---|
| [0] | 1 | 2 | 3 |
| [1] | 4 | 5 | 6 |

**Exercise:** Initialize A[2][3] with same values through for loop

- How a 2D array is stored?



# Multi-dimensional Arrays

- Accessing 2D arrays using Pointers:


name of the array is pointer

```
cout<<A[1];
cout<<*(A+1);
```

- How does 2D arrays and pointers relate:

A[0] A[1]

| A[0][0] | A[0][1] | A[0][2] | A[1][0] | A[1][1] | A[1][2] |
|---|---|---|---|---|---|

# Multi-dimensional Arrays

- Now… If I want access A[1][2] via pointer, what will I write?

| | |
|---|---|
| First Solution | *(A[1] + 2) |
| Second Solution | *(*(A+1) + 2) |

- You can view **3D** or **nD** array the same way, i.e., int A[2][2][2]