

Department of Computer Science and Engineering

# **Data Structures and Object-Oriented Design**

(CSE - 2050)

#### **Hasan Baig**

Office: UConn (Stamford), 305C email: <a href="mailto:hasan.baig@uconn.edu">hasan.baig@uconn.edu</a>

20

#### CSE-2050 - Data Structures and Object-Oriented Design

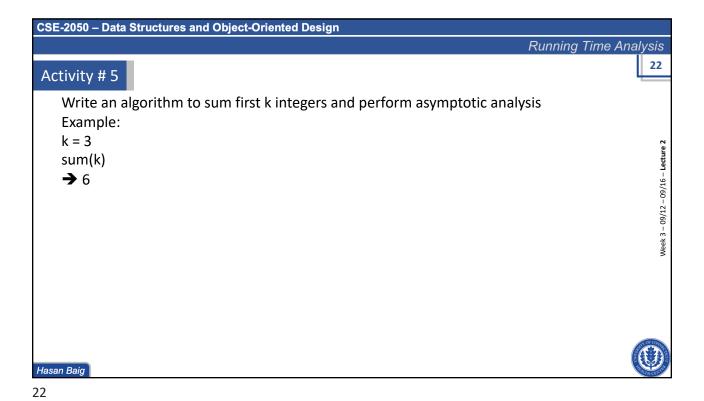
Recap

21

#### **Quick Recap**

- How to measure algorithm's performance in terms of timings?
  - Some issues with measuring timings
- We studied timings of an algorithm to find duplicates (duplicates 1) in input data
  - Discussed why duplicates 1 algorithm was not efficient
  - · Improved and compared performance of both algos
- Asymptotic Analysis and Atomic operations
- Examples of operations in lists
- Activity # 5 for creating an algorithm to calculate the sum of K integers

Hasan Baig



Activity # 5 Solution

Write an algorithm to sum first k integers and perform asymptotic analysis

1 def sum\_calc(k):
2 total = 0
3 for i in range(1, k+1):

4 total += i
5 return total
6
7 print(sum\_calc(3))
8 print(sum\_calc(6))
9 print(sum\_calc(7))
10 print(sum\_calc(100))
hasanbaig@HBMAC Prac % python3 sum\_calc.py
6
21
28
5050

2k + 2

CSE-2050 – Data Structures and Object-Oriented Design

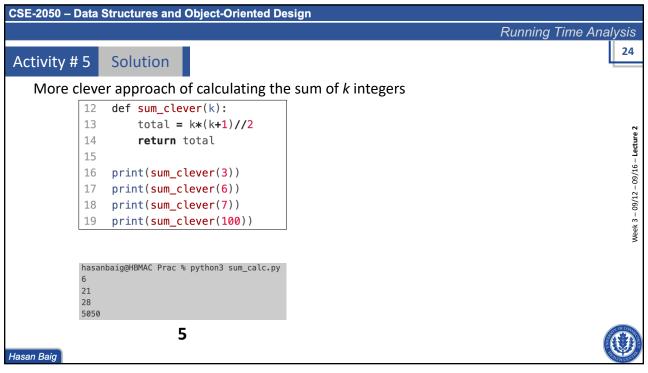


Running Time Analysis

23

23

Hasan Baig



24

```
CSE-2050 – Data Structures and Object-Oriented Design
                                                                         Running Time Analysis
                                                                                           25
Asymptotic Analysis
   Lets perform asymptotic analysis of duplicate_2 program
     10
          def duplicates_2(L):
     11
               n = len(L)
     12
               for i in range(1,n):
                                                                    n^2/2 - n/2 + 3
     13
                    for j in range(i):
     14
                        if L[i] == L[j]:
     15
                             return True
     16
               return False
Hasan Baig
```

### CSE-2050 – Data Structures and Object-Oriented Design

Running Time Analysis

# **Asymptotic Analysis**

26

- Only higher order terms significantly affects a function
- In asymptotic analysis, we drop lower order terms and constants
  - Consider only higher order terms which grows faster with input size, n

$$5n^2 + 3n + 2$$

Hasan Baig

26

#### CSE-2050 – Data Structures and Object-Oriented Design

Running Time Analysis

# **Big-O Notation**

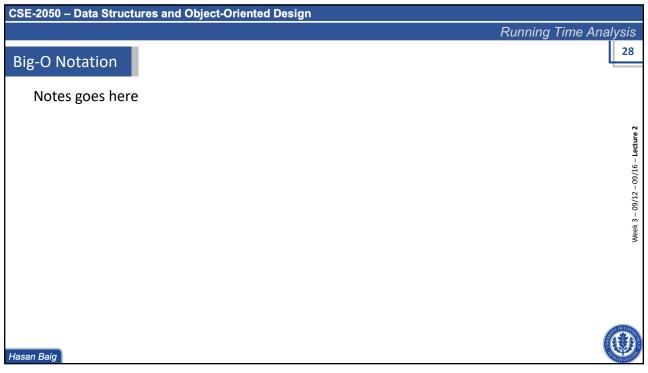
27

• The formal mathematical definition which allows us to ignore lower order terms and constants is called **Big-O** notation

3 - 09/12 - 09/16

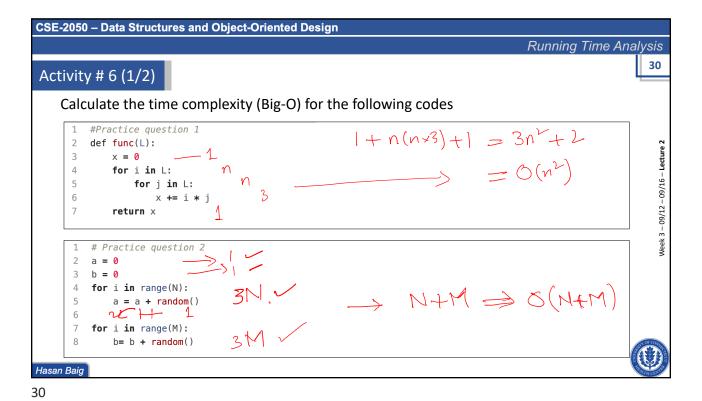
Hasan Baig





28

```
CSE-2050 – Data Structures and Object-Oriented Design
                                                                           Running Time Analysis
Big-O Notation
    Example:
          def duplicates_2(L):
      10
      11
               n = len(L)
               for i in range(1,n):
     12
                                                                     n^2/2 - n/2 + 3
      13
                    for j in range(i):
      14
                         if L[i] == L[j]:
                                                                          O(n<sup>2</sup>)
     15
                              return True
     16
               return False
Hasan Baig
```



CSE-2050 - Data Structures and Object-Oriented Design Running Time Analysis 31 Activity # 6 (2/2) N = 10Calculate the time complexity (Big-O) for the following codes 1 #Practice question 3 for i in range(N): → N 0010 for j in reversed(range(i,N)): 4 a = a + i + j;1 # Practice question 4 i = 1 while i <= n: print(i) 4 i \*= 2 Hasan Baig