# Data Structures and Algorithms

# INFO 6205

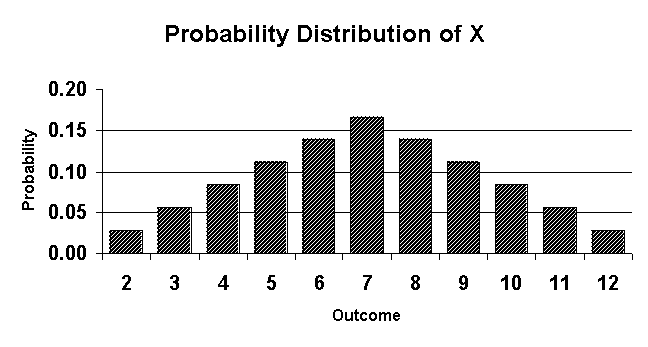
# Homework 2

# Due: May 23, 2020

Put all your java, compiled class files and documentation files into a zip file named Homework2.zip and submit it via the dropbox on the blackboard before the END of due date.

Put your name on all .java files. There will be a short quiz on this homework.

1. Suppose the customers enter a Bank has the following histogram:



a) What is Random variable?

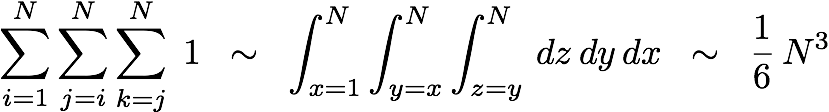
b) What is probability distribution for this distribution throwing two dices?

c) What is Mean and Standard Deviation of Probability distribution?

d) Explain the observed statistics for a Bank system.

2. Write the code that results to the following running time. The 3-Sum Triple loop has

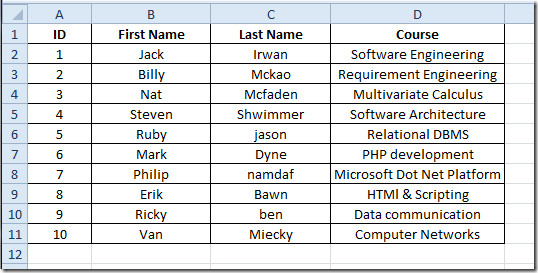
the following running time estimate. Do Not prove Math. Just want explaining the math,

what does it represents and why the result is 1/6 N^3

3. For the LinkedList implementation of Queue example I discussed in class, write a TestLinkedListQueue class to test enqueue, dequeue,, isEmpty and tother operations as needed.

4. Describe the Array Implementation of Queue with one example discussed in class. You need to provide a sample data and walk through the enqueue and dequeue, and other operations as necessary and manage the head and trail pointers.

5. Consider the following data:



Build **Queue** with LinkedList implementation and Array implementation:

a) Create file “input.txt” with this data

b) Read input.data into an an ArrayList.

c) Create Queue with LinkedList implementation

d) Write Node data structure of your input data

e) Queue must support all operations of queue

f) Write a Test program to test your linked implementation of Queue:

—enqueue 3 elements into queue

—dequeue 2 elements from queue

—enqueue all elements into queue

—dequeue all elements from queue

—enqueue all elements into queue

—print queue with the goal:

i) reverse order ii) original order as was first read into array list

g) Compile and Run your program

h) what is Queue LinkedList time-complexity?

i) Repeat (a)—(g) with build **Queue fixed Array** Implementation

j) what is Queue Fixed Array time-complexity?

k) What are the consequences of oversizing or undersizing fixed array size?

6. Consider the following Algorithm to Evaluate Infix Expressions for these expressions:

A \* B / C + (D + E - (F \* (G / H)))

4 + 8 \* 6 - 5 / 3 - 2 \* 2 + 2

(1 + 3 + ( ( 4 / 2 ) \* ( 8 \* 4 ) ))

(4+8) \* (6-5) / ((3-2) \* (2+2))

a) Apply Algorithm to evaluate these Infix expressions, show step-by-step

b) Write Java code for the algorithm to evaluate Infix expressions

c) Compile and Run

Algorithm:

Iterate through given expression, one character at a time

If the character is an operand, push it to the operand stack.

If the character is an operator,

If the operator stack is empty then push it to the operator stack.

Else If the operator stack is not empty,

If the character’s precedence is greater than or equal to the precedence of the stack top of the operator stack, then push the character to the operator stack.

If the character’s precedence is less than the precedence of the stack top of the operator stack then do Process (as explained above) until character’s precedence is less or stack is not empty.

If the character is “(“, then push it onto the operator stack.

If the character is “)”, then do ***Process*** (as explained above) until the corresponding “(” is encountered in operator stack. Now just pop out the “(“.

Once the expression iteration is completed and the operator stack is not empty,

do ***Process*** until the operator stack is empty.  The values left in the operand stack is our

final result.