ASSIGNMENT INSTRUCTIONS

1. Assignment 03: **75 points w/ 0 E.C. points**
2. Due Date & Time: **03-11-2020 at 11:55 PM**

WHAT TO SUBMIT

1. Assignment Report
2. Code

HOW TO SUBMIT AND THE RULES TO FOLLOW

- Submit via iLearn, the Assignment Submission section
- Please refer to Assignment 01 for the Assignment Guidelines
- Please follow the Assignment Report Template
- Please follow the Course Policy on Student Conduct and Academic Honesty

PERFORMANCE TRACKER		
ASMT	GRADE	YOUR GRADE
ZOOM	05	
01	20	
02-PREPARATION	25	
02	75	
03	75	
TOTAL	200	

A: 90-100% **B**: 80-89% **C**: 70-79% **D**: 60-69% **F**: 0-60%
The course grader provides feedback to your assignments on iLearn.

ABOUT

Assignment 02 was a thorough practice of Object-Oriented Programming and common utilities. We completed reviewing CSC 210 and gained a strong foundation in programming and in project management. Time management is an important component of project management. We learned to use available resources for success. We learned the fundamentals of problem solving.

Assignment 03 provides us with opportunities to explore new data structures, starting with the two primary data structures, and data structure programming techniques. Mastering these sets of skills will enable us to learn the more complex data structures.

- Please download and use the **starter code provided**: http://csc220.ducta.net/Assignments/**Assignment-03-Code**.zip
- For all the assignment parts, we **can add** code. We **cannot change** and **cannot remove** the provided code. Thank you.

PART A – The Linked Bag, **20 points**

Please implement class **`LinkedBag`** focusing on method **`removeAllOccurrences`**. This method removes all occurrences of the given entries from a bag. Our program's output must be **identical** to the following same output:

NO User Input

```
=== LINKED BAG 220 JAVA ============================================================
[+] Creating a CSC220 LinkedBag...
[+] Adding.... these items to the bag:    A _ _ G Bb A _ u n o A o d Bb A A l l
[>] The bag now contains 18 string(s):    l l A A Bb d o A o n u _ A Bb G _ _ A
[+] Creating... a 2D test array with the below contents:
                A A A A A A
                B A Bb B Bb B
                C B _ A
                n u l l
[+] Removing 2D test array items from the bag...
 [-] Converting 2D array to 1D...
 [-] Removing duplicates in 1D array...
 [>] The final 1D array now contains:  A B Bb C _ n u l
 [-] Removing the final 1D array items from the bag...
[>] The bag now contains 4 string(s):  G o o d
=== LINKED BAG 220 JAVA ============================================================
```

PART B – Stack, **15 points**

Please **add code** to the provided code. Our program tests whether an input string is a palindrome. Our program output must be **identical** to this sample output.

A **palindrome** is a string of characters (a word, phrase, or sentence) that is the same regardless of whether we read it forward or backward. For example, *Race car* is a palindrome. So is *A man, a plan, a canal: Panama.* More about palindromes: *http://www.palindromelist.net/*. We ignore spaces

Interactive

```
[>>] Enter a string (or a ! to exit): Csc
 [+] Yes. "Csc" IS a palindrome!
[>>] Enter a string: CSC 220 - Data Structures
 [-] No. "CSC 220 - Data Structure" is NOT a palindrome!
[>>] Enter a string: Ah, Satan sees Natasha!
 [+] Yes. "Ah, Satan sees Natasha!" IS a palindrome!
[>>] Enter a string: Amy, must I jujitsu my ma?
 [+] Yes. "Amy, must I jujitsu my ma?" IS a palindrome!
[>>] Enter a string: A man, a plan, a canal: Panama.
 [+] Yes. "A man, a plan, a canal: Panama." IS a palindrome!
[>>] Enter a string: Are Mac 'n' Oliver evil on camera?
 [+] Yes. "Are Mac 'n' Oliver evil on camera?" IS a
   palindrome!
[>>] Enter a string: !
[<<] Thank you!
```

and punctuations. Please do not worry about ellipsis, n-dash, and m-dash, and case. Please see *https://simple.wikipedia.org/wiki/Punctuation* .

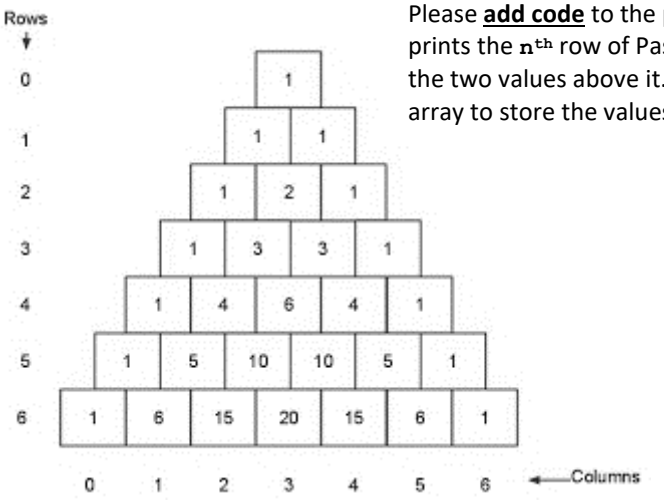
PART C – Recursion, **15 points**

In the language of an alien race, all words take the form of Blurbs. A Blurb is a Whoozit followed by one or more Whatzits. A Whoozit is the character 'x' followed by zero or more 'y's. A Whatzit is a 'q' followed by either a 'z' or a 'd', followed by a Whoozit. Please **add code** to the provided code to complete a recursive program that generates random Blurbs in this alien language. Our program must behave in the same manner as the sample run does:

Interactive

```
Enter a number of blurbs: 7
Blurb #1: xyyyqdxyyyyyyyqdxyy
Blurb #2: xqdxyqzxyyyyyqzx
Blurb #3: xyqdxyyyy
Blurb #4: xqzxqzx
Blurb #5: xyqdxyyyyyy
Blurb #6: xyqdxqzxqzxqzxqdxyyyy
Blurb #7: xqzx
```

PART D – Recursion, **10 points**



Please **add code** to the provided code to complete a recursive program that determines and prints the n^{th} row of Pascal's Triangle (as shown on the left). Each interior value is the sum of the two values above it. Our program output must be **identical** to the sample out. *Hint:* use an array to store the values on each line.

Interactive

```
Enter a row of Pascal's Triangle: 0
1
Enter a row of Pascal's Triangle: 1
1 1
Enter a row of Pascal's Triangle: 2
1 2 1
Enter a row of Pascal's Triangle: 4
1 4 6 4 1
Enter a row of Pascal's Triangle: 6
1 6 15 20 15 6 1
Enter a row of Pascal's Triangle: 11
1 11 55 165 330 462 462 330 165 55 11 1
Enter a row of Pascal's Triangle: !
Adiós!
```

Text in Red

PART E – The Efficiency of Algorithms, **15 points**

1. Show how you count the number of operations (not only basic operations) required by the algorithm to the right, **5 points**:

```
int i, n = 5, sum = 5;
for (i = 5; i < 3 * n; i++) {
   sum *= n + i * 7 + 37;
}
```

It is OK to do this part on a sheet of paper then snapshot it. Please include the screenshot in our assignment report. Microsoft Office Lens, Adobe Scan, and Google Lens are good phone applications for this purpose.

***Coding is recommended but not required**. Please make sure our handwriting is readable to our grader*.

2. Consider Loop A and Loop B in the box to the right, **5 points**:

   Although Loop A is $O(n)$ and Loop B is $O(n^2)$, Loop B can be faster than Loop A for small values of n. Design and code a creative experiment to find a value of n for which Loop B is faster.

   Please submit both our code and our discussion.

```
// Loop A
for (i = 1; i <= n; i++)
   for (j = 1; j <= 10000; j++)
      sum = sum + j;

// Loop B
for (i = 1; i <= n; i++)
   for (j = 1; j <= n; j++)
      sum = sum + j;
```

3. Repeat question E.2 but use Loop C in place of Loop B, **5 points:**

   Please submit both our code and our discussion.

```
// Loop C
for (i = 1; i <= n; i++)
   for (j = 1; j <= n; j++)
      for (k = 1; k <= n; k++)
         sum = sum + k;
```

Happy Coding!