La Salle College
Algorithms & Programming (420AP1AS)

Michelle M. Khalifé
Assignment 4 – arrays, 2D arrays

#1. [warm-up]

a. Declare and initialize an array *num[10]* of type *int*, such that the value at index *i* is *i+1.*
b. Modify your code to replace the values in the array with their square. E.g., if num[i] = 5, then num[i] = 25
c. Return the sum of values in the array

#2. **[final grades]**

a. Write a program that reads grades from an array and computes the following: class average, max grade, and min grade. The program needs to ask the user how many students there are in the class in order to create & fill the array.
b. Create a separate array that captures the distribution of these grades and lets the user know the percentage of students that falls in the different ranges.

#3. [reverse an array]

a. Write a program that reads characters in an array *A* and prints them in reverse.
b. Modify the program to copy the content of array *A*, in reverse order, in an array B. Display the content of both arrays.
c. Read the sequence of characters in both arrays from [0] to [n-1]. Is it the same?

#4. [arrays @ w3resources] Exercises 32-36, 38, 42, 54, and 58.

#5. [merge]

a. Write a program that reads two sorted arrays *A[m]* and *B[n]*, in parallel, and outputs their elements in ascending order. For simplicity, start with *m=n* and then modify the program so *m≠ n*.
b. Modify the program to store the elements in a third array *C[m+n].* Don't forget to account for duplicates.

#6. [for-patterns] Redo the *for*-patterns in Assignment 3. Instead of printing the pattern, save it into a 2D array.

#7. [grid]

a. Prompt the user for two integers *m* and *n* and create an *m x n* grid. The value at the (i, j) position is *i + mj.* Fill the grid with the appropriate value by traversing rows first, and cols second.

0	5	10	15	20	25
1	6	11	16	21	26
2	7	12	17	22	27
3	8	13	18	23	28
4	9	14	19	24	29

b. Iterate the grid by traversing cols first, and rows second. Here, let the value at position (i,j) be k++, where k is initialized to 0 prior to looping. Print the grid. Is it the same as the grid above?

#8. [sum of rows & cols]

Prompt the user for 9 number that you will read in a 2D array. You can do so in more than one way: either one number at a time *or* 3 at a time. Then compute the sum of each row and column and output the elements in the following format:

ARRAY			**ROW SUM**
1	2	3	6
4	5	6	15
7	8	9	24
COLUMN SUM			
12	15	18	

#9. [TIC TAC TOE]

Write a program that allows a user to play tic tac toe. The program should ask for moves alternating between player X and player O. Alternatively, the second player can be the machine. The program displays the game positions (1-9) and when a player selects a position, their move is entered, and the new board configuration is displayed. The board on the left is the original board. The board on the right shows the configuration after 4 moves.

1	2	3
4	5	6
7	8	9

X	X	O
4	5	6
O	8	9

The game ends when a player has 3 Xs or 3 Os spanning across the board (horizontally, vertically, or diagonally), or there are no more moves that can be made.

#10. [pascal Δ]

In the pattern below, boundaries aside, every row can be computed from the previous row. The element at position (i,j) is the sum of the elements at position (i-1, j-1) and (i-1, j). Reproduce and print the pattern.

```
1
1   1
1   2   1
1   3   3   1
1   4   6   4   1
1   5   10  10  5   1
1   6   15  20  15  6   1
```