Spring 2019

Fundamentals of Computer Programming
Lab 6 – Dynamic Memory

1-

a. Write a function int NegateArray(double *Arr, int size);
The function changes the sign of each element in array ***Arr*** and returns the count of zeros elements in the array.

b. Write a program to test the function.

c. Change the function header to be and notice the effect of each

i. int NegateArray(const double *Arr, int size); then
ii. int NegateArray(double const *Arr, int size); then
iii. int NegateArray(double *const Arr, int size);

2- Write a function "DuplicateArray" that takes an array of double (a double pointer) and array size. The function **creates** and **returns** a new array whose elements are the duplicate of the original array elements.
Example:
Original array = { 1, 2, 5, 3, 10 } ➔ Returned array = {2, 4, 10, 6, 20}
Decide parameters and return type of the function.

3- Write a function "ReplicateMe" that takes an array of double (a double pointer) and array size. The function replicates the input array and doubles its size.
Example:
Original array before call = { 1, 2, 5 }
Original array after call = {1, 1, 2, 2, 5, 5}
Decide parameters and return type of the function.

4- Write a program to test function ReplicateMe as follows

a. Ask the user to enter N (no. of elements)
b. Allocate an array (say A) dynamically of size N
c. Ask the user to enter N values and store them in the A
d. Print array A.
e. Call ReplicateMe function and print array A after function call
f. Delete the allocated array

5- Write a function void RemoveChar(char *&S, int &size, char X); that takes an array of characters, its size and a character **X**. The function removes all occurrences of char **X** from array S.
Example:
If function is called with S = "exercise", size =8, and X = 'e'
Then after function finishes: S = "xrcis", and size =5
Note: After removing all occurrences of X, if there are no more characters left in S, so S should be set to NULL.