Looping/Repetition/Iteration

# Demonstration of Questions 7-11 is required

Looping – this is the third control structure we have worked with in this course. Each control structure brings some advantage to the programmer. This one allows the coder to do more and at the same time write less code. This is possible by processing a block of code statements more than once. C# provides 4 looping constructs.

The do-while is the most intuitive of the looping constructs. The syntax of the do-while structure is:

int counter = 0;  
do  
{  
 Console.WriteLine(counter);  
 counter++;  
}while(counter < 5);

initializer  
do  
{  
 statement 1;  
 statement 2;  
}while(«boolean expression»);

Lab Exercise on Looping Using a do-while loop write the following programs:

1. To display five asterisks on a single line. Each asterisk must be separated by a space. You must use a do-while loop to solve this program
2. To display the numbers 1 … 5 on separate lines
3. To display the numbers 1 … 20 on separate lines
4. To display the numbers 1 … 20 on separate lines with an asterisk after each multiple of 5. (multiples asterisk)
5. To display the numbers 10 … 40 on separate lines with an asterisk before each multiple of 7. (asterisk multiples)
6. To display numbers 20 … 60 on separate lines skipping the multiple of 3. i.e. You must print a blank line for each multiple.
7. To repeatedly prompt for a number and sum it. When the sum just exceeds 100, stop the prompting and display the sum at the end. You must not display the sum while the user in typing in numbers. (You do not need a counter but you will need some way of terminating the loop)
8. Same as the previous question but additionally displays the average of the numbers when the loop terminates. You will need a counter but not as a loop terminator.
9. To display the sum of all the multiples of 3 between 1000000 and 2000000.   
   (Answer = **499, 999, 500, 000**). If you suspect that a value might be larger than 2 billion (as is the case here), then the type of sum should be a long and not an int.   
   [Hint: like the **C** (currency) and the **F** (decimal) format specifiers, there is also a **N** specifier]
10. A conversion table of Celsius to Fahrenheit temperature. The table must start with 100 Celsius and end at 0 Celsius with decrements of 10. (Fahrenheit = 9/5 \* Celsius + 32). Your table must have a suitable header and the values in the table must be right-align like the output of question 11.
11. A machine purchased for $28,000 depreciates at a rate of $4,000 a year for seven years. Write and run a C# program that computes and displays a depreciation table for seven years. The table should have the form:

END-OF-YEAR ACCUMULATED  
YEAR DEPRECIATION VALUE DEPRECIATION  
---- ------------ ------------ ------------  
 1 4000 24000 4000  
 2 4000 20000 8000  
 3 4000 16000 12000  
 4 4000 12000 16000  
 5 4000 8000 20000  
 6 4000 4000 24000  
 7 4000 0 28000