In Class 7 – `for` Loops

In this in-class assignment, you write some loops in C++, mostly `for` loops . This assignment along with the previous in-class assignment should help you understand loops and when it is best to use a `for` loop and when to use a `while` loop.

```
/*  Write C++ code to do each of the
 *  following.
 */
```

1. Create a new project, start with main() from Hello World, with the #include and using statements. Your code must include all of these loops. If you skip one, then the input and output will not work with the tests on Web-CAT. That is because my tests will be entering input that you do not have the code for. Therefore my input will erroneously go to the next step, and will therefor produce incorrect output. For example, if you skip #5, then my test-code input for #5 will be used by #6, which would product incorrect output. Print labels like #2:, #3:, etc. with each part. See sample output below.
2. Create a for loop to print the numbers 1 to 10.
3. Add a new for loop to the code to print the even numbers 0 to 100, just the even numbers.
4. Add a new for loop to prompt for a positive number and **print** all the numbers between 0 and that number; use a for loop.
5. Add a new for loop to prompt for a positive number, then prompt and read in that many `int`s and add them up. Print the total. Change the for loop to a while loop that does the same thing. You must leave only the while loop; either delete or comment out the for loop.
6. No prompt necessary, but you may use one. Add another loop that reads int numbers and adds them to a total, looping until a negative number is entered. In other words, read and sum up int positive numbers until a negative number is entered, then print the total. Note that the first number may be negative so that 0 is printed.
7. Prompt for a positive int number. Then prompt, read in, and add up int values until that first number is entered again. Print the total. Do not add in the 2 values of the beginning & ending int. For example, if the user enters -1, then you add the entered values until you get another -1, but you do not add in either -1 value.

Use a for loop for the rest of these

8. Print 5 lines with 1 asterisk per line.
```
*
*
*
*
*
```
9. Print 5 asterisk on 1 line. Print and endl at the end of the line.
```
*****
```

10. Prompt for a positive number and print a line with that many asterisk on a line. For example,
```
Input a Number: 7
*******
```

11. Using 2 for loops, one inside the other, and a cout that only prints 1 asterisk, print a hill shaped triangle like this:
```
Input a Number: 5
*
**
***
****
*****
```

12. Now change the for statements to print it upside down:
```
Input a Number: 6
******
*****
****
***
**
*
```

13. Now change the code to print this: (notice there are fewer lines, (n+1)/2 lines)
```
Input an ODD Number: 9
*
***
*****
*******
*********
```

14. Now change the for statements to print this:
```
Input an ODD Number: 7
*******
*****
***
*
```

Now you are ready for the next Programming assignment!

See below for sample Input & Output, but first…
NOTE that Web-CAT will only provide hints that the output for one of the parts is wrong. For example, "hint: Output for #6 is not correct." , means your output doesn't match the string "#6:\nEnter positive numbers to add up until a negative number is entered: total: 7\n#7". Notice this does not include the input, but it does include every space, colon, and carriage return. In other words, you MUST make your output match the format of the output below, including pound signs (hash tags), spaces, lack of spaces, carriage returns and blank lines (or lack of them).

Also note, Part 11 is tested again a string like " Input a Number: ******\n*****\n****\n***\n**\n*\n"

Sample Input & Output

```
#2:
1
2
3
4
5
6
7
8
9
10
#3:
0
2
4
6
8
10
12
```

… I have deleted some of the numbers so this output isn’t so long.

```
96
98
100
#4:
Input a Number to print up to: 4
0
1
2
3
4
#5:
How many numbers to add up: 4
3
2
11
7
total: 23
#6:
Enter positive numbers to add up until a negative number is entered:
2
4
-1
```

```
total: 6
#7:
Input a Sentinel Number: 12
2
3
4
12
total: 9
#8:
*
*
*
*
*

#9:
*****

#10:
Input a Number: 5
*****

#11:
Input a Number: 6
*
**
***
****
*****
******

#12:
Input a Number: 7
*******
******
*****
****
***
**
*

#13:
Input an ODD Number: 9
*
***
*****
*******
```

```
*********
```

#14:

```
Input an ODD Number: 15
***************
*************
***********
*********
*******
*****
***
*
```

Also... This is a type of question you can expect on quizzes and tests:

```
int index = 0;
// Continue looping infinitely until internal condition is met.
while (true)
{
    int value = ++index;
    // Check to see if limit it hit.
    if (value > 5)
    {
        cout << "While-loop break\n\n";
        break;
    }
    // You can add another condition.
    if (value > 100)
    {
        cout << "Never hit\n\n";
      return 1; // or exit(1);
    }
    // Write to the screen on each iteration.
    cout << "While-loop statement\n";
}
```

What is output by this code?