CMSC337: Algorithms: Design & Practice

Lab#1: Language Standards, Reference, Reference Compilers, and finding odd numbers

For two of your preferred programming languages, identify the following:

1. What is the current version of the language?

C:_____   Go:_____   Java:_____   Python:_____

2. For your preferred two languages, what is the authoritative language reference (printed, or online):

C:

Go:

Java:

Python:

3. For your preferred two languages, what compiler(s) are available on CS machines?

C:_____   Go:_____   Java:_____   Python:_____

For your two preferred languages, what IDE(s), if any, are available on CS machines?

C:_____   Go:_____   Java:_____   Python:_____

4. Consider the function defined below:

$$isOdd(n) = \begin{cases} true & if \, n \, is \, odd \\ false & o/w \end{cases}$$

5. You can use the modulo operator **(%)** to test if a number is odd. That is, **n** is an odd number if **(n % 2)** equals 1. Thus, the function **isOdd(n)** can be defined using the algorithm shown below:

function isOdd(n: integer) returns true/false
    if (n % 2) equals 1
        return true
    else
        return false

6. Next, code the function **isOdd()** using the algorithm described in (5) in at least **two** programming languages using the latest versions identified in (1) that can be compiled using the compilers identified in (3) above . The function should return a boolean value. Only in case a boolean type is not available in a language, you may use an integer result where 1 stands for true and 0 for false.

7. Write a program that uses **isOdd()** defined above, using the algorithm shown below:

   for each value, say i in 0..5
       print out the value returned by the expression isOdd(i)

   Format the output so it looks as follows:
   0 false
   1 true
   2 false
   3 true
   4 false
   5 true
   Note, and confirm, that the program is correctly identifying 1, 3, and 5 as odd numbers (i.e. prints true), and that 0, 2, and 4 are correctly identified as not being even numbers (i.e., prints false). Ensure that you have tested all two of your programs in this manner.

8. Next, modify the programs to use the following algorithm:

   for each value, say v in -5..5
       print out the value returned by the expression isOdd(v)

   Run all programs and note down your results. You will have a total of 2x11 cases. Two programming languages, each program producing 11 outputs.

9. Study the 22 cases closely and ensure that all outputs are correct. Guess what, they will not be! For the language whose program gave incorrect results, study the language reference(s) identified in (2) above to investigate the cause.

10. Based on your analysis, revise the algorithm from step 5 so it produces correct results. Then revise your programs to reflect your revised algorithm.

11. Test, using the algorithm from (8) again. Confirm that the results are correct!

12. **[Extra Credit]** Study if any or all the programming languages provide bitwise operations. You can use the bitwise AND operation for an alternate, and faster, implementation of **isOdd()** as below:

function isOdd(n: integer) returns true/false
  return (n BITWISE_AND 1) != 0

Test all your programs again using the algorithm in (8) to confirm that you have a correct implementation.

13. Write a short essay based on this little computational excursion. The target audience of your essay should be seasoned computer science students (like yourselves). You essay should serve as an informative tutorial that succinctly, and in an engaging manner, teaches the reader about the importance of language standards, compilers, pitfalls, traps, and safe coding techniques.

   Note: You do not need to include all the details from this lab, only the most relevant ones. The essay should be no longer than 2-3 printed pages (single-line spacing, 1-inch margins all around).

   What to submit:

   a. Your essay.

   b. Associated with your essay should be two appendices:

      i. An Appendix showing your answers to (1) through (3) above.

      ii. An Appendix with the showing all of your code (there should be 4 programs, more if you did the extra credit).