SENG1000 – C/C++ PROGRAMMING

FOCUSED ASSIGNMENT 3 - CALLING FUNCTIONS AND MAKING DECISIONS

OVERVIEW

Write a program that calls a function to get a number from the user and displays whether the number is even or odd.

GENERAL COURSE OBJECTIVES ADDRESSED IN THIS ASSIGNMENT

- Use comments appropriately, according to industry-accepted criteria.
- Declare and initialize variables.
- Use programmer-defined functions.
- Use parameters and return values.
- Use input and output functions.
- Use if/else statements.

ACADEMIC INTEGRITY AND LATE PENALTIES

- Link to Academic Integrity Information
- Link to Late Policy

EVALUATION

- The evaluation of this assignment will be done as detailed in the Marking lecture from Week 2.

PREPARATION

- View Week 2 and 3 videos.

REQUIREMENTS

USER INPUT:

- Use the getNum() function below to obtain user input. Do not change the getNum() function.
- Prompt the user before calling getNum(). Do **not** prompt the user within the getNum() function.
- Only one number is obtained. The program does not loop. Your mark will be capped at 60 if your program loops.

ISODD():

- Create and use a function called isOdd() with a parameter (the number) and return values (1 if the number is odd, 0 if the number is even OR use a bool or boolean data type, your choice) to determine if the number is odd.
    - There is an operator covered in the Operators lecture that will help you immensely with isOdd(). Use it.
    - Do **not** do output within isOdd().
- Call isOdd() from main() and use the return value to determine whether or not the user entered an odd or even number.
    - If it is odd, display the number followed by " is odd". Otherwise, display the number followed by " is even". Then use "\n" to go to a new line.
- Create a prototype for isOdd().

GETNUM():

You must use the following function to get input:

```
#pragma warning(disable: 4996)      // required by Visual Studio

int getNum(void)
{
/* the array is 121 bytes in size; we'll see in a later lecture how we can
improve this code */
char record[121] = {0}; /* record stores the string */
int number = 0;
    /* NOTE to student: brace this function consistent with your others */

    /* use fgets() to get a string from the keyboard */
    fgets(record, 121, stdin);

    /* extract the number from the string; sscanf() returns a number
     * corresponding with the number of items it found in the string */
    if( sscanf(record, "%d", &number) != 1 )
    {
        /* if the user did not enter a number recognizable by
         * the system, set number to -1 */
        number = -1;
    }
    return number;
}
```

- This function should be put at the end of your source file. You can use copy-and-paste.
- Make sure that you create a function comment and prototype for getNum(). Reformat the bracing and indentation of getNum to match your own style.
- You don't have to understand the code inside of getNum in order to use it (it uses C features we haven't covered yet).

COMMENTING AND INDENTATION:

- Make sure that you use a header comment as described in the Commenting lecture. It goes before the #include line in your source file.
- Create function comments for getNum() and isOdd().
- Inline comments should use the two principles discussed in the Commenting lecture.
- Make sure that your indentation and bracing is correct and consistent (including for getNum) throughout your entire program.

OTHER REQUIREMENTS:

- All variables must be declared within functions (i.e. it must not use global variables (covered later in the Scope and Style lecture)).
- Appropriate programming style as discussed in lecture and in the Course Notes must be used.
- **Do not have any input or output except as required by this assignment.**
- Do not clear the screen (this requirement is true for all assignments).
- It is assumed that you will adhere to all course requirements detailed in the Course Notes readings so far in the course. **This requirement holds for all subsequent assignments.**

CHECKLIST REQUIREMENTS

- Create a requirements checklist. This should contain the specific requirements from this assignment as well as any relevant requirements that have been covered in lecture or that are found in the SET Coding Standards or SET Submission Standards. Do it in whatever form you wish. Hand in your completed checklist in PDF form as checklist.pdf. Not having this checklist will result in a cap of 80 on your mark.

FILE NAMING REQUIREMENTS

- You must call your source file f3.cpp.
- You must call your checklist checklist.pdf.

SUBMISSION REQUIREMENTS

- Do not hand in any other source files besides those mentioned in the File Naming Requirements.
- Follow the instructions in the SET Submission Standards and the lecture on Submitting Assignments to submit your program. Submit both files to the correct Assignment folder.
- Once you have submitted your files, make sure that you've received the eConestoga e-mail confirming your submission. Do not submit that e-mail (simply keep it for your own records until you get your mark).

ADDITIONAL INFORMATION

- You can assume that the user will not enter negative numbers, non-numbers, or excessively long input.
- Both prototypes go before any function definitions. main() should not have a prototype.