1. Recursive functions:
   a. a. Write a recursive function with the following prototype that returns the sum of all the digits of an integer:

      ```
      int sum_of_digits(int);
      ```

      For example, both -23 and 23 will return 5.
   b. Write a recursive function to get an n^{th}-level approximation to the golden mean ratio,

      $$\tau = \frac{1}{1+\frac{1}{1+\frac{1}{1+\cdots}}}$$

      For example, the 1nd level approximation would be $\tau \approx \frac{1}{1+1} = 0.5$
   c. Write a main program to test your functions

2. Write a program to complete the following tasks:
   a. Print a message “Do you want to play (Y/N)?”
      i. If user inputs ‘Y’ or ‘y’
         1. Ask user for a positive integer
            a. If the input is not a positive integer
               i. print out a message “try again “
               ii. Ask the user for the input again
         2. Otherwise
            a. If the number is even, divide it by 2
            b. If it’s odd, multiple by 3 and add 1
            c. Repeat the steps 1&2 until the result is 1.
            d. Print out message “it took 3 steps to reduce 8 to 1” in the case where the input is 8.
      ii. Otherwise, skip to the end of the program and exit normally
   b. Repeat step a.

   Note: try to use switch statements wherever you can instead of if…else if…

3. BONUS (extra 25pts): (*optional*)
   Write a recursive function, *move_rings*, that shows the solution to the following puzzle:

   You have 3 stacks of rings. Each ring is smaller than the one it sits. To start out, all rings are at stack A. Challenge: move all rings from stack A to stack C, subject to these constraints: (a) you can only move one ring at a time; (b) you only move any ring to either stack A, B, or C; (c) you can place a ring only on top of a larger ring.

   Each time a ring is moved, your function will print out a message about the move. For example, if you are moving a ring from stack B to stack A, print out the message “move from B to A”.

   Hint:

   - you know the base case solution where there is just one ring
   - the parameters of the function should have the information on
     - the number of rings to be moved;
     - the stack the rings are to be moved from;

    - the stack the rings are to be moved to;
    - the spare stack

```
void move_rings(int n, char from, char to, char spare);
```

In main() function, get a user input for the number of rings, then call your *move_rings* function. In the case of n=2, your output should look like this:

```
How many rings in stack A? 2
move from A to B
move from A to C
move from B to C
```

Submit all your source files along with the screen shots of their output on line.