KIE1008: Programming 2
Week 2: Basic Concepts – Classes Interface and Implementation

1. Find the syntax errors in the following class definition.

```
1    class discover()
2    {
3    public:
4       set(string, int, double);
5       void print() const;
6       discover() const;
7       discover(string, int, double);
8       set(string, int, double);
9    private;
10      string type;
11      int l;
12      double w;
13   }
```

2. Design and implement a class `dayType` that implements the day of the week in a program. The class `dayType` should store the day, such as Sun for Sunday. The program should be able to perform the following operations on an object of type `dayType`:
   a) Set the day.
   b) Print the day.
   c) Return the day.
   d) Return the next day.
   e) Return the previous day.
   f) Calculate and return the day by adding certain days to the current day.
      For example, if the current day is Monday and we add 4 days, the day to be returned is Friday. Similarly, if today is Tuesday and we add 13 days, the day to be returned is Monday.
   g) Add the appropriate constructors.

3. Write a program for the class `clock` to execute the following commands:
   a) Define a class object `myClock` with parameters hours = 8, minutes = 12, and seconds = 30. Print out the values on the screen.
   b) Define another class object `yourClock` with the parameters hours = 12, minutes = 35, and seconds = 45. Print out the values on the screen.
   c) Change the parameters in `myClock` using `setTime()` function with the parameters hours = 3, minutes = 48, seconds = 52. Print out the new values of `myClock`.
   d) Compare the values between `myClock` and `yourClock`. Print out the result.
   e) Assign the values of `myClock` into `yourClock` using assignment operator (=). Print out the new values on the screen.
   f) Compare again the values between `myClock` and `yourClock`. Print out the result.
   g) Increase `myClock` to an hour faster. Print out the result.

   The following outputs are expected:

```
My clock is 08:12:30
Your clock is 12:35:45
Now, my clock is changed to 03:48:52 Our clocks have different time.
If your clock is changed to 03:48:52
Our clocks have same time.
My clock is increased an hour to 04:48:52
```

4. Design a class that implements the basic properties of a die. Consider the definition of the following class die.

```
class die
{
public:
    die();
    //Default constructor
    //Sets the default number rolled by a die to 1
    void roll();
    //Function to roll a die.
    //This function uses a random number generator to randomly
    //generate a number between 1 and 6, and stores the number
    //in the instance variable num.
    int getNum() const;
    //Function to return the number on the top face of the die.
    //Returns the value of the instance variable num.
private:
    int num;
};
```

   a) Write the definitions of the functions to implement the operations for the class `die`.
   b) Write a program that uses the class `die` to roll 10 dice. (Use an array of size 10 to implement 10 dice.) The program prompts the user to enter the desired sum and the number of times the dice are to be rolled.
      The program outputs the number of times the desired sum was rolled. Test run your program to roll the 10 dice 10000 times with the desired sum indicated by the user.

5. (Tic-Tac-Toe) Write a program that allows two players to play the tictac- toe game. Your program must contain the class `ticTacToe` to implement a `ticTacToe` object. Include a 3-by-3 two-dimensional array, as a `private` member variable, to create the board. The constructor should initialize the empty board to all zeros. Wherever the first player moves, place a 1 in the specified square. Place a 2 wherever the second player moves. Each move must be to an empty square. After each move, determine whether the game has been won or is a draw. If needed, include additional member variables.
   Include some of the operations on a `ticTacToe` object are printing the current board, getting a move, checking if a move is valid, and determining the winner after each move. Add additional operations as needed.