Object Oriented Programming & Design
CS 244 - 001
Department of Physics and Computer Science
Medgar Evers College
Exam 1 Make-Up

Instructions:

- **Create the class ‘`Clock.h`’ that contain a header guard and includes the libraries *iostream*, *string*, *sstream*, and *iomanip*.**
- **Your submissions must be submitted to the GitHub repository in the Exam01 directory.**
- **Cheating of any kind is prohibited and will not be tolerated.**
- **Violating or failing to follow any of the rules above will result in an automatic zero (0) for the lab.**

TO ACKNOWLEDGE THAT YOU HAVE READ AND UNDERSTOOD THE INSTRUCTIONS ABOVE, PRINT YOUR NAME AND THE DATE ON BOTH THIS SHEET AND THE BLUE BOOK

Name: ____________________________________ **Date:** ________________

Grading

Task	Maximum Points	Points Earned
00	0.05	
01	0.15	
02	0.15	
03	0.15	
04	0.25	
05	0.25	
06	0.25	
07	0.25	
08	0.25	
09	0.25	
10	0.25	
11	0.25	
12	0.25	
13	0.25	
14	0.25	
15	0.25	
16	0.25	
17	0.25	
18	0.25	
19	0.25	
20	0.25	
21	0.25	
Total	5	

Within a namespace named *oope* (task 00), define the class *Clock* that contains

1. a private integer array field named *times* with size 4 that represents the time hour, time minute, alarm hour, and alarm minute components, respectively.

2. a private integer field named *volume*.

3. a private Boolean array field named *switches* with size 3 that represents format (true for 12 hour, false for 24 hour), alarm state (true for on, false for off), and mode (true for alarm view, false for time view), respectively.

4. a private static constant string array field named *meridiems* initialized to {`"AM","PM"`}.

5. a private static integer field named *count* initialized to zero.

6. a public default constructor that initializes *times*, *volume*, and *switches* to {0, 0, 6, 0}, 30, and {false, false, false}, respectively, and increments *count* by 1.

7. a public copy constructor.

8. a public assignment operator.

9. a public destructor that decrements *count* by 1.

10. a public void method named `mode()` that takes no parameters and toggles the mode.

11. a public void method named `alarm()` that takes no parameters and toggles the alarm state.

12. a public void method named `format()` that takes no parameters and toggles the format.

13. a public static method named `clocks()` that takes no parameters and returns *count*.

14. a public constant getter method named `hour()` for hour of the mode.

15. a public constant getter method named `minute()` for minute of the mode.

16. a public constant getter method named `level()` for *volume*.

17. a public void method named `adjustTime()` takes two Boolean parameters and adjusts the specified component of the mode in the indicated direction given that the component (first parameter) implies hour if true and minute if false, and the direction (second parameter) implies increment one unit if true and decrement one unit if false.

18. a public void method named `adjustVolume()` takes a Boolean parameter and adjusts *volume* in the indicated direction if possible given that the direction (the parameter) implies increment one unit if true and decrement one unit if false.

19. a public Boolean constant method named `sounds()` that takes no parameter and returns true if alarm state is true and the time is greater than or equal to alarm, otherwise, it returns false.

20. a public string constant method `toString()` that takes no parameters and returns a string in the format

$$\begin{cases} \texttt{"(}v\texttt{) }h\texttt{:}m\ z\texttt{"} & \text{if format is 12 hour} \\ \texttt{"(}v\texttt{) }h\texttt{:}m\texttt{"} & \text{if format is 24 hour} \end{cases}$$

    where v is `'T'` if the mode is time view or `'A'` if the mode is alarm view, h is the hour of the mode as two digits, m is the minute of the mode as two digits, and z is the times corresponding meridiem.

21. a friend ostream operator that returns an output in the same format as `toString()`.

Extra Credit

Within *clock* define the string method named `zone()` that takes a string parameter and returns time in 24-hour `toString()` format at the timezone (the parameter) provided given that the current timezone is `EST`. The acceptable timezones are `EST`, `CST`, `MST`, `PST`, `AKT`, `HAT`, and `AST`. (Only acceptable if main assignment receives at least a 4).