Homework 1: Classes and Objects in C++

Task 1: Create a Student class in the Student.cpp file with the following information:

Student
- ID: string - Name: string - Score: double
+ Student() + Student(id: string, name: string) + setID(id: string): void + getID(): string + setName(name: string): void + getName(): string + setScore(score: double): void + getScore(): double + getGrade(): string

Explanation:

- Student class stores the information of a student.
- ID, Name, Score are private data members.
- Student() is the no-arg constructor.
- Student(id: string, name: string) is the parameterized constructor used to initialize the values to ID and Name.
- setID, getID, getName, setName, setScore, getScore are getters and setters.
- In setter setScore need to validate the score from **score** param. The score is guaranteed to be greater than 0 and less than or equal to 10 (if score < 0, assign 0 to score, if score > 10, assign 10 to score).
- The getGrade function member will return letter grade which is calculated by following table:

Score	Letter Grade
>= 9.0	A
>= 8.0 && < 9.0	B+
>= 7.0 && < 8.0	B
>= 6.0 && < 7.0	C+
>= 5.0 && < 6.0	C
>= 4.5 && < 5.0	D+
>= 4.0 && < 4.5	D
< 4.0	F

Write a main function to create 3 instances of Student class by using constructor and output the information and letter grade to the screen.

Task 2: Create a Point class in the Point.cpp file with the following information:

Point
- x: double - y: double
+ Point() + Point(x: double, y: double) + setX(x: double): void + getX(): double + setY(y: double): void + getY(): double + setPoint(x: double, y: double): void + distance(x: double, y: double): double + distance(another: Point): double

Explanation:

- Point class stores the information of coordinates of a point in a two-dimensional coordinate system.
- x, y are private data members.
- Point() is the no-arg constructor, then x, y assign to 0.
- Point(x: double, y: double) is the parameterized constructor used to initialize the values to x and y.
- setX, getX, getY, setY, setPoint are getters and setters.
- The distance(x: double, y: double) function member will return distance between the current and the point (x, y)
- The distance(another: Point) function member will return distance between the current and another object (also the object of Point class).

$$Distance\ formula: AB = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}$$

Write a main function to create 2 instances of Point class by using constructor and out the distance between two points by using both two distance function members.

```cpp
#include <iostream>
#include "Point.cpp"

using namespace std;

int main() {
    Point p1(1.5, 6.7);
    Point p2(2.8, 3.2);
    cout << p1.distance(p2) << endl;
    cout << p1.distance(2.34, 7.8) << endl;
    return 0;
}
```

Task 3: Create a Math class in the Math.cpp file with the following information:

Math
- PI: double
+ getPI(): double + abs(a: double): double + add(a: double, b: double): double + sub(a: double, b: double): double + min(a: double, b: double): double + min(arr[]: double, n: int): double + max(a: double, b: double): double + max(arr[]: double, n: int): double + pow(x: double, n: int): double + avg(a: double, b: double): double + avg(arr[]: double, n: int): double

Explanation:

- Math class stores the information of PI and several static function members.
- PI is private data member and equal 3.14159.
- abs will return absolute value of a
- add will return result of a + b
- sub will return result of a – b
- min(a: double, b: double) will return minimum value between a and b
- max(a: double, b: double) will return maximum value between a and b
- avg(a: double, b: double) will return average value between a and b
- pow(x: double, n: int) will return result of x^n
- min(arr[]: double, n: int) will return minimum value of array **arr**
- max(arr[]: double, n: int) will return maximum value of array **arr**
- avg(arr[]: double, n: int) will return average value of array **arr**

Write a main function to implement all static function members.

```cpp
#include <iostream>
#include "Math.cpp"

using namespace std;

int main() {
    double a = -6, b = 8, c[10] = { 1, -2, 8, -5, 12, 9, 7, -4, 3, -6 };
    int n = 5;
    cout << "PI: " << Math::getPI() << endl;
    cout << "Absolute of A: " << Math::abs(a) << endl;
    cout << "Addition of A and B: " << Math::add(a,b) << endl;
    cout << "Subtraction of A and B: " << Math::sub(a) << endl;
    cout << "Minimum of A and B: " << Math::min(a) << endl;
    cout << "Maximum of A and B: " << Math::max(a) << endl;
    cout << "Average of A and B: " << Math::avg(a) << endl;
    cout << "A to the power of n is: " << Math::pow(a,n) << endl;
    cout << "Minumum of array C: " << Math::min(c) << endl;
    cout << "Maximum of array C: " << Math::max(c) << endl;
    cout << "Average of array C: " << Math::avg(c) << endl;
    return 0;
}
```

Task 4: Create a Book class in the Book.cpp file with the following information:

Book
- title: string - author: string - quantity: int
+ Book() + Book(title: string) + Book(title: string, author: string) + getTitle(): string + setTitle(title: string): void + getAuthor(): string + setAuthor(author: string): void + getQuantity(): int + setQuantity(quantity: int): void

Explanation:

- Book class stores the information of a book and quantity.
- title, author, quantity are private data members.
- Book() is the no-arg constructor.
- Book(title: string) is the parameterized constructor used to initialize the values to title. The author is empty, and quantity is 0.
- Book(title: string, author: string) is the parameterized constructor used to initialize the values to title and author. The quantity member is 0.
- setTitle, getTitle, getAuthor, setAuthor, getQuantity, setQuantity are getters and setters. setQuantity function should ensure that quantity is greater than 0.

Write a main function to implement this class. Create a loop to implement borrow process.

```cpp
#include <iostream>
#include "Book.cpp"

using namespace std;

int main() {
    Book b1("C++ Primer, 5th Edition","Stanley B. Lippman");
    Book b2("Object-Oriented Programming Simplified","Hari Mohan Pandey");
    Book b3("Design Patterns in Modern C++","Dmitri Nesteruk");

    b1.setQuanity(5);
    b2.setQuanity(2);
    b3.setQuanity(4);

    string title;
    int quantity;
    while(true){
        cout << "Please input book title: ";
        getline(cin,title);
        cout << "Please input quantity: ";
        cin >> quantity;

        // TODO CODE
    }

    return 0;
}
```

Requirements:

- If the input title is not match, output warning: "Book not found".
- If the input title is match, check the quantity of that book.
- If the quantity is enough, output the success message and decrease amount from that book.
- If the quantity is not enough, output the warning message: "The quantity is not enough. Please try again!"

Task 5: Create a Time class in the Time.cpp file with the following information:

Time
- hour: int - minute: int - second: int
+ Time() + Time(hour: int, minute: int, second: int) + getHour(): int + setHour(hour: int): void + getMinute(): int + setMinute(minute: int): void + getSecond(): int + setSecond(second: int): void + add(second: int): void + sub(second: int): void + getTime(): string + compare(another: Time): int

Explanation:

- Time class stores the information of hour, minute and second.
- hour, minute, second are private data members.
- Time() is the no-arg constructor. Set hour, minute and second to 0.
- Time(hour: int, minute: int, second: int) is the parameterized constructor used to initialize the values to hour, minute and second.
- setHour, getHour, getMinute, setMinute, getSecond, setSecond are getters and setters. Ensure: $0 \leq$ hour ≤ 23, $0 \leq$ minute ≤ 59, $0 \leq$ second ≤ 59.
- add(second: int) will add amount of seconds in to this time.
- sub(second: int) will subtract amount of seconds in to this time.
- getTime function member will return the time in format "HH:MM:SS".
- compare(another: Time) will return -1 if the current time is less than another time, return 1 if the current time is greater than another time, 0 if the both time objects are equal.

Write a main function to create 2 instances of Time class by using constructor. Adjust time on the time object by using getters and setters. Implement comparison on these two time objects.

```cpp
#include <iostream>
#include "Time.cpp"

using namespace std;

int main() {
    Time t1(23,9,18);
    Time t2(3,56,23);
    cout << "Time t1: " << t1.getTime() << endl;
    cout << "Time t2: " << t2.getTime() << endl;

    t1.setHour(5);
    t1.setMinute(12);
    t1.setMinute(45);
    cout << "Time t1: " << t1.getTime() << endl;

    t1.sub(4365)
    t2.add(3817);

    if (t1.compare(t2) < 0) {
        cout << "t1 is less than t2" << endl;
    } else if(t1.compare(t2) < 0) {
        cout << "t1 is greater than t2" << endl;
    } else {
        cout << "t1 is equal to t2" << endl;
    }
    return 0;
}
```