Tasmita Tanjim
19723 student id.

Replit link: https://replit.com/@TASMITA-TANJIMT/CS360HW2-1

1. Modify class *GradeBook* as follows:
   a. Include a second-string data member that represents the course instructor's name.
   b. Provide a *set* function to change the instructor's name and a *get* function to retrieve it.
   c. Modify the constructor to specify course name and instructor name parameters.
   d. Modify function *displayMessage* to output the welcome message and course name, then the *string "This course is presented by: "* followed by the instructor's name.

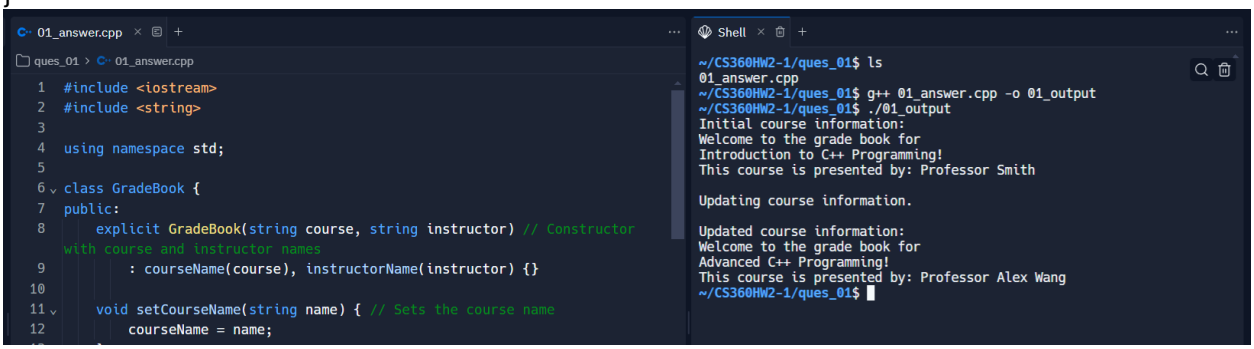
Use your modified class in main function that demonstrates the class's new capabilities.

```
#include <string>

class GradeBook{
 public:
  explicit GradeBook( std::string );  // constructor initialize courseName
  void setCourseName( std::string );  // sets the course name
  std::string getCourseName() const;  // gets the course name
  void displayMessage() const;    // displays a welcome message

 private:
  std::string courseName; // course name for this GradeBook
}; // end class GradeBook

#include <iostream>
using namespace std;

GradeBook::GradeBook( string name ):courseName( name ){}

void GradeBook::setCourseName( string name ){
  courseName = name;
}

string GradeBook::getCourseName() const{return courseName;}

void GradeBook::displayMessage() const{
  cout << "Welcome to the grade book for\n" << getCourseName()
     << "!" << endl;
}
```

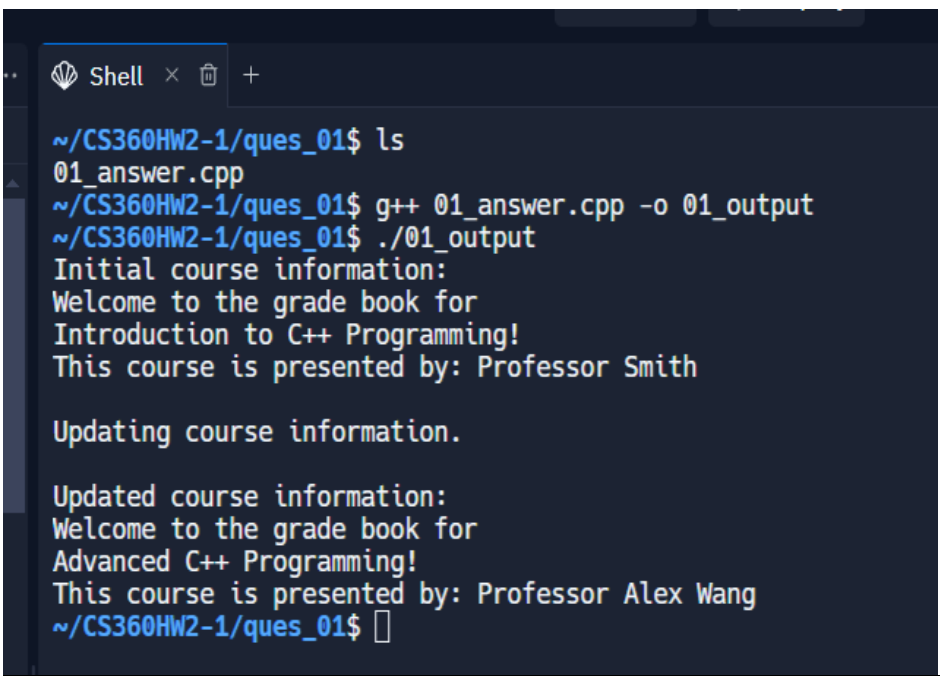
ANSWER:

```
#include <iostream>
#include <string>

using namespace std;

class GradeBook {
public:
   explicit GradeBook(string course, string instructor) // Constructor with course and instructor
names
      : courseName(course), instructorName(instructor) {}

   void setCourseName(string name) { // Sets the course name
      courseName = name;
   }

   string getCourseName() const { // Gets the course name
      return courseName;
   }

   void setInstructorName(string name) { // Sets the instructor's name
      instructorName = name;
   }

   string getInstructorName() const { // Gets the instructor's name
      return instructorName;
   }

   void displayMessage() const { // Displays a welcome message including the instructor's name
      cout << "Welcome to the grade book for\n" << getCourseName() << "!" << endl;
      cout << "This course is presented by: " << getInstructorName() << endl;
   }

private:
   string courseName; // Course name for this GradeBook
   string instructorName; // Instructor's name for this GradeBook
};

int main() {
   // Creating an instance of GradeBook with initial course and instructor names
   GradeBook gradeBook("Introduction to C++ Programming", "Professor Smith");

   // Displaying initial class information
   cout << "Initial course information:" << endl;
   gradeBook.displayMessage();
```

```
    // Changing the course name and instructor name
    cout << "\nUpdating course information." << endl;
    gradeBook.setCourseName("Advanced C++ Programming");
    gradeBook.setInstructorName("Professor Alex Wang");

    // Displaying updated class information
    cout << "\nUpdated course information:" << endl;
    gradeBook.displayMessage();

    return 0;
}
```



OUTPUT:



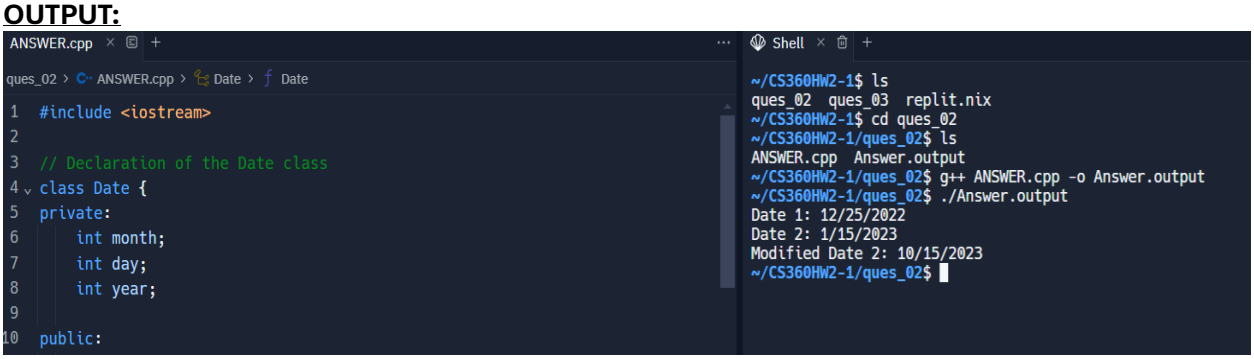
2. Create a class called *Date* that includes three pieces of information as data members--a month (type *int*), a day (type *int*) and a year (type *int*). Your class should have a constructor with three parameters that uses the parameters to initialize the three data members. Assume that the values provided for the year and day are correct but ensure that the month value is in the range *1-12*; if it isn't, set the month to *1*. Provide a *set* and a *get* function for each data member. Provide a member function *displayDate* that displays the month, day and year separated by forward slashes (/). Write a test program that demonstrates class *Date's* capabilities.

CODE:

```
#include <iostream>

// Declaration of the Date class
class Date {
private:
  int month;
  int day;
  int year;

public:
  // Constructor with parameter validation for the month
  Date(int m, int d, int y) : day(d), year(y) {
    setMonth(m); // Use setMonth to ensure month is set correctly according to class rules
  }

  // Set functions for each data member with validation where necessary
  void setMonth(int m) {
    month = (m >= 1 && m <= 12) ? m : 1; // Ensure month is within the valid range
  }

  void setDay(int d) {
    day = d; // Set day without validation as per requirements
  }

  void setYear(int y) {
    year = y; // Set year without validation as per requirements
  }

  // Get functions for each data member
  int getMonth() const {
    return month;
  }

  int getDay() const {
    return day;
```

```
    }

    int getYear() const {
        return year;
    }

    // Function to display the date in mm/dd/yyyy format
    void displayDate() const {
        std::cout << month << "/" << day << "/" << year << std::endl;
    }
};

// Test program to demonstrate the capabilities of class Date
int main() {
    // Create Date objects
    Date date1(12, 25, 2022); // Valid date
    Date date2(13, 15, 2023); // Invalid month, should reset to 1

    // Display both dates
    std::cout << "Date 1: ";
    date1.displayDate();
    std::cout << "Date 2: ";
    date2.displayDate();

    // Modify date2's month to a valid value and display again
    date2.setMonth(10);
    std::cout << "Modified Date 2: ";
    date2.displayDate();

    return 0;
}
```

OUTPUT:



3.While exercising, you can use a heart rate monitor to see that your heart rate stays within a safe range suggested by your trainers and doctors. According to the American Heart Association (AHA) (www.americanheart.org/presenter.jhtml?identifier=4736), the formula for calculating your maximum heart rate in beats per minute is 220 minus your age in years. Your target heart rate is a range that is 50-85% of your maximum heart rate. [Note: These formulas are estimates provided by

the AHA. Maximum and target heart rates may vary based on the health, fitness and gender of the individual. Always consult a physician or qualified health care professional before beginning or modifying an exercise program.]. Create a class called HeartRates. The class attributes should include the person's first name, last name and date of birth (consisting of separate attributes for the month, day and year of birth). Your class should have a constructor that receives this data as parameters. For each attribute provide set and get functions. The class also should include a function getAge that calculates and returns the person's age (in years), a function getMaxiumumHeartRate that calculates and returns the person's maximum heart rate and a function getTargetHeartRate that calculates and returns the person's target heart rate. Since you do not yet know how to obtain the current date from the computer, function getAge should prompt the user to enter the current month, day and year before calculating the person's age. Write an application that prompts for the person's information, instantiates an object of class HeartRates and prints the information from that object—including the person's first name, last name and date of birth—then calculates and prints the person's age in (years), maximum heart rate and target-heart-rate range.

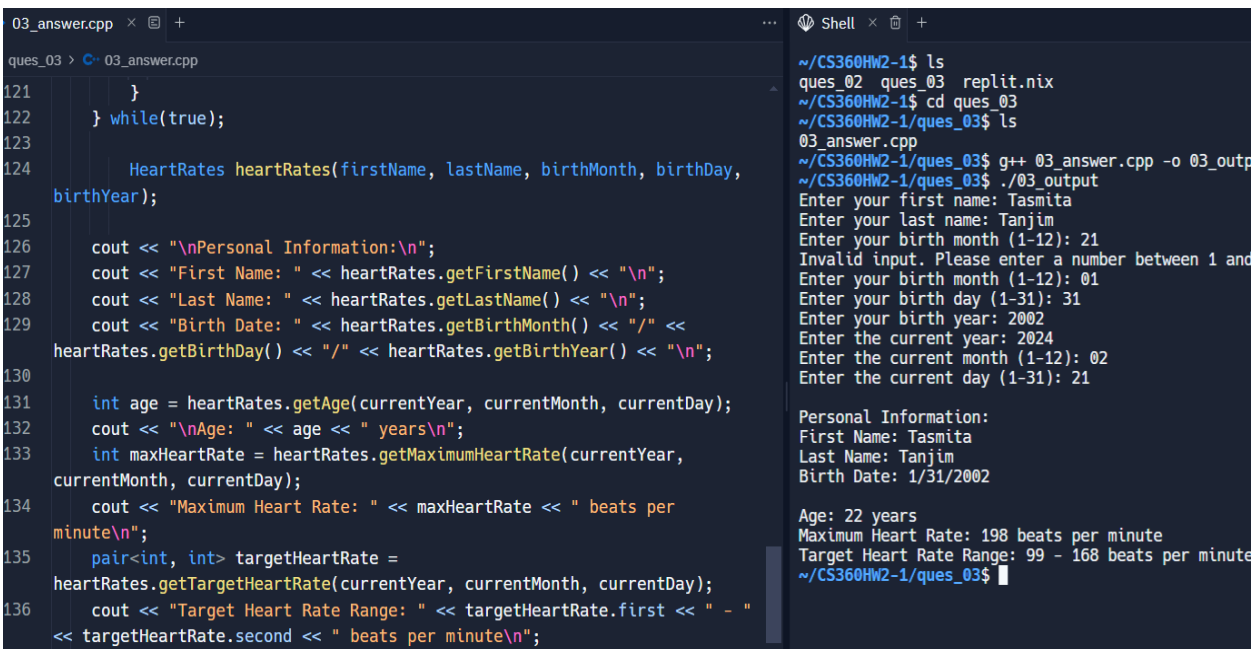
CODE:

```
#include <iostream>
#include <string>
#include <utility> // For std::pair
#include <limits>  // Required for std::numeric_limits

using namespace std;

class HeartRates {
private:
   string firstName;
   string lastName;
   int birthMonth;
   int birthDay;
   int birthYear;

public:
   HeartRates(string fn, string ln, int bm, int bd, int by)
      : firstName(fn), lastName(ln), birthMonth(bm), birthDay(bd), birthYear(by) {}

   // Getters
   string getFirstName() const { return firstName; }
   string getLastName() const { return lastName; }
   int getBirthMonth() const { return birthMonth; }
   int getBirthDay() const { return birthDay; }
   int getBirthYear() const { return birthYear; }

   // Calculate age
   int getAge(int currentYear, int currentMonth, int currentDay) const {
      int age = currentYear - birthYear;
      if (currentMonth < birthMonth || (currentMonth == birthMonth && currentDay < birthDay)) {
         age--;
```

```
    }
    return age;
  }

  // Calculate maximum heart rate
  int getMaximumHeartRate(int currentYear, int currentMonth, int currentDay) const {
    return 220 - getAge(currentYear, currentMonth, currentDay);
  }

  // Calculate target heart rate range
  pair<int, int> getTargetHeartRate(int currentYear, int currentMonth, int currentDay) const {
    int maxHeartRate = getMaximumHeartRate(currentYear, currentMonth, currentDay);
    return make_pair(static_cast<int>(maxHeartRate * 0.5), static_cast<int>(maxHeartRate *
0.85));
  }
};

int main() {
  string firstName, lastName;
  int birthMonth, birthDay, birthYear;
  int currentYear, currentMonth, currentDay;

  cout << "Enter your first name: ";
  cin >> firstName;
  cout << "Enter your last name: ";
  cin >> lastName;

  // Input validation for birth month
  do {
    cout << "Enter your birth month (1-12): ";
    cin >> birthMonth;
    if(cin.fail() || birthMonth < 1 || birthMonth > 12) {
      cout << "Invalid input. Please enter a number between 1 and 12.\n";
      cin.clear();
      cin.ignore(numeric_limits<streamsize>::max(), '\n');
    } else {
      break;
    }
  } while(true);

  // Input validation for birth day
  do {
    cout << "Enter your birth day (1-31): ";
    cin >> birthDay;
    if(cin.fail() || birthDay < 1 || birthDay > 31) {
      cout << "Invalid input. Please enter a number between 1 and 31.\n";
      cin.clear();
      cin.ignore(numeric_limits<streamsize>::max(), '\n');
```

```
    } else {
        break;
    }
} while(true);

// Input validation for birth year
do {
    cout << "Enter your birth year: ";
    cin >> birthYear;
    if(cin.fail() || birthYear < 1900) { // Adjust according to valid birth year range
        cout << "Invalid input. Please enter a valid year.\n";
        cin.clear();
        cin.ignore(numeric_limits<streamsize>::max(), '\n');
    } else {
        break;
    }
} while(true);

// Input validation for current date
cout << "Enter the current year: ";
cin >> currentYear;
do {
    cout << "Enter the current month (1-12): ";
    cin >> currentMonth;
    if(cin.fail() || currentMonth < 1 || currentMonth > 12) {
        cout << "Invalid input. Please enter a number between 1 and 12.\n";
        cin.clear();
        cin.ignore(numeric_limits<streamsize>::max(), '\n');
    } else {
        break;
    }
} while(true);

do {
    cout << "Enter the current day (1-31): ";
    cin >> currentDay;
    if(cin.fail() || currentDay < 1 || currentDay > 31) {
        cout << "Invalid input. Please enter a number between 1 and 31.\n";
        cin.clear();
        cin.ignore(numeric_limits<streamsize>::max(), '\n');
    } else {
        break;
    }
} while(true);

    HeartRates heartRates(firstName, lastName, birthMonth, birthDay, birthYear);

cout << "\nPersonal Information:\n";
```

```
    cout << "First Name: " << heartRates.getFirstName() << "\n";
    cout << "Last Name: " << heartRates.getLastName() << "\n";
    cout << "Birth Date: " << heartRates.getBirthMonth() << "/" << heartRates.getBirthDay() << "/" << heartRates.getBirthYear() << "\n";

    int age = heartRates.getAge(currentYear, currentMonth, currentDay);
    cout << "\nAge: " << age << " years\n";
    int maxHeartRate = heartRates.getMaximumHeartRate(currentYear, currentMonth, currentDay);
    cout << "Maximum Heart Rate: " << maxHeartRate << " beats per minute\n";
    pair<int, int> targetHeartRate = heartRates.getTargetHeartRate(currentYear, currentMonth, currentDay);
    cout << "Target Heart Rate Range: " << targetHeartRate.first << " - " << targetHeartRate.second << " beats per minute\n";

    return 0;
}
```



<u>OUTPUT:</u>

I have also used loops in case the user gives the wrong input. It will again ask for valid input from the user.

```
~/CS360HW2-1$ ls
ques_02  ques_03  replit.nix
~/CS360HW2-1$ cd ques_03
~/CS360HW2-1/ques_03$ ls
03_answer.cpp
~/CS360HW2-1/ques_03$ g++ 03_answer.cpp -o 03_output
~/CS360HW2-1/ques_03$ ./03_output
Enter your first name: Tasmita
Enter your last name: Tanjim
Enter your birth month (1-12): 21
Invalid input. Please enter a number between 1 and 12.
Enter your birth month (1-12): 01
Enter your birth day (1-31): 31
Enter your birth year: 2002
Enter the current year: 2024
Enter the current month (1-12): 02
Enter the current day (1-31): 21

Personal Information:
First Name: Tasmita
Last Name: Tanjim
Birth Date: 1/31/2002

Age: 22 years
Maximum Heart Rate: 198 beats per minute
Target Heart Rate Range: 99 - 168 beats per minute
~/CS360HW2-1/ques_03$
```