Programming Exam 1

CMPE 126 Section 06 Lab Version 1

Date: 03/08/2024 9:00am-11:50am

This version of the exam is ONLY for students whose SJSU ID end in 0, 1, 2, 3, and 4.

General guidelines

- This exam will be for 2.5 hours only. Please use the last 20 minutes to complete your submission.
- Please submit your best work BEFORE 11:50am
- You may use external sources **with citation**

Academic Dishonesty

The following will be considered actions of academic dishonesty and will be reported without further discussion.

- Copying code:
  - You are NOT permitted to share code or reuse someone else's work.
- Discussing with others:
  - You are NOT permitted to discuss with your classmates or anyone else about the lab.

Remember to submit your OWN work and **cite all your sources**.

Expected outcome from this exam

The exam will test your skills and understanding of the following concepts:

- C++ programming skills
- Problem-solving skills
- Object-oriented programming concepts - mainly encapsulation, abstraction, and polymorphism.
- Operator overloading (mainly arithmetic, logical, relational, << and >>)
- Arrays

Exam in the next page…

The problem

You are creating a software to track boxes for a moving company. In this exam, you will:

1. Create a class called `imperialWeight` with `pounds` and `ounces` fields. Note that the `ounces` field can have only integers 0 – 15 and the `pounds` field can only have positive integers.
2. Overload the `<<` operator to output a weight in the format "`pounds` lbs, `ounces` oz".
   E.g., 4 lbs 6 oz should read 4 in the field `pounds`, 6 in the field `ounces`.
3. Create a class called `Box` with fields `name` and `weight`. Name should be of type `string` and `weight` should be of type `imperialWeight`.
4. Overload `>`, `<`, `<=`, `>=` to compare two objects of type `imperialWeight`.
5. Test the functionality of the overloaded operators in a `progExam1.cpp` file containing main.
6. Create an array of ten Box objects called Room. Initialize it with the details provided in the table below.

Box name	Weight in the format `pounds` lbs, `ounces` oz
Books	25 lbs, 12 oz
Table Left	5 lbs, 4 oz
Table Right	10 lbs, 5 oz
Paintings	7 lbs, 1 oz

7. Design and implement an algorithm to print the heaviest box's name and weight.
8. BONUS 1:
   a. Throw an error if the format of the weight is incorrect.
   b. Overload operator+ for the imperialWeight class to add two imperialWeight objects.
   c. Test the functionality of the overloaded operator+ in progExam1.cpp.
9. BONUS 2 – Write functions to implement the following functionalities:
   a. addBox – add a Box object to the array.
   b. totalWeight – the total weight of all the boxes in the array.

Rubric

Passing criteria:

1. imperialWeight class
2. Operator<< overload
3. Box class

Higher grades: C level

1. Code compiles and runs
2. Array operations

Higher grades: B level

1. Operator >, <, >=, <= overload
2. Find heaviest

Higher grades: A level

1. Bonus 1
2. Bonus 2

Submission instructions

Submit the following files in a zip file:

- imperialWeight.h
- imperialWeight.cpp
- Box.h
- Box.cpp
- progExam1.cpp
- Notes: Include all your references in this file. Feel free to include any implementation details.

Name your zip file ProgrammingExam1_XXXYYYXXX.zip, where XXXYYYXXX is your SJSU ID. E.g., if your SJSU ID is 111000111, your submission will be called ProgrammingExam1_111000111.zip.