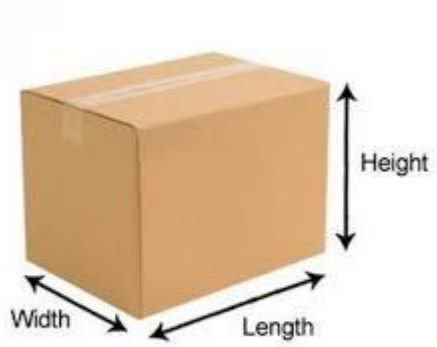
ITU Computer Engineering Department
BLG252E Object Oriented Programming
3rd Homework

Due Date: May 6, 2016 23:00



In this homework, you will implement **Box, Toy** and **Book** classes for a shopping company. The box class may carry any kind of objects such as toys, books and other boxes. However, the type of objects which will be carried out with in a specific box should be defined at declaration time. Some rules must be taken into consideration and if the given rules are not met, exceptions must be thrown.

- Design and implement these classes in C++ while **avoiding code repetition** as much as possible for all classes and providing **data hiding**.
- Test code is given to guide the design of your classes.
- All the dynamic data members should be declared as **private**.
- Box, toy, and book classes should contain a label, weight, and dimensions (length and width) of the related object (no need for height value, you may assume that it is unlimited). Toy class also has a Boolean flag which determines if it contains batteries or not.
- Box class also has a maximum allowed weight parameter.
- Box class should be able to calculate the total weight of itself by considering the objects that it contains.
- You should successfully **deallocate** all of the allocated memory before termination of your program.
- **Add** method of the box class should dynamically increase the size of an object array. The following constraints should be checked and an exception should be thrown for any attempts to add new objects.
    - The total weight of contained object including the box may not exceed the maximum allowed weight for the box.
    - The dimensions of the contained object should be equal or smaller than those of the box. You can assume that the height of the box is infinite and each object can be placed on top of the previous one. The objects can be rotated 90 degrees along the z axis, which will swap the length and width of it. (so that an object of w=5 l=8 fits into a box with w=8 and l=7)
- Make sure that GNU C++ compiler (g++) compiles your project and the application runs in Linux smoothly.
- **You are <u>not</u> allowed to use the standard template library (STL) or make any changes in the test code.**
- Your implementation must be compatible with the given test program (main.cpp) and should generate a similar output as the given sample (sampleoutput.txt).

A test program is given below. It illustrates the usage of all the methods and operators you will implement. Your implementation must be compatible with this test program.

```
#include <iostream>
#include <cstdlib>
#include <ctime>
#include <string>
#include "box.h"
#include "box.cpp"
#include "toy.h"
#include "book.h"
using namespace std;
int main() {
    //Toy Constructor: label, weight, length, width, containsBattery
    Toy toy1("Lego: Heroes Batman",1.2,0.2,0.3,false);

    //Box Constructor: weight, length, width, maximumAllowedWeight
    Box<Toy> toyBox(0.1,1.0,1.0,7.0);

    try {
        toyBox.add(toy1);
        cout << toyBox[0]<< endl;

        // trying to get the element at(1)
        // should give an error
        cout << toyBox[1] << endl;
    } catch (const string & err_msg) {
        cout <<"### ERROR ### "<< err_msg << endl;
    }

    Toy toy2("Lego: Star Wars Death Star",1.7,0.5,0.3,false);
    toyBox.add(toy2);
    cout<<endl<<toyBox<<endl;

    Toy *toyArray= new Toy[2];
    toyArray[0]=  Toy("Hot Wheels: Turbine Twister Track Set",1.4,0.3,0.3,false);
    toyArray[1]=  Toy("Lego: Heroes Superman",1.6,0.3,0.3,false);

    Box<Toy> toyBox2=toyBox;
      toyBox2.add(toyArray,2);

    delete [] toyArray;

    try {
        Toy toy3("Lego: Heroes Superman",1.0,0.7,1.1,false);
        // trying to add a toy bigger than the box
        // should give an error
        toyBox.add(toy3);
    } catch (const string & err_msg) {
        cout <<"### ERROR ### "<< err_msg << endl;
    }
    try {
        Toy toy3("Hot Wheels: Speedtropolis Playset",6,0.7,0.8,false);
        // trying to add a toy heavier than the box's remaining available weight
        // should give an error
        toyBox.add(toy3);
    } catch (const string & err_msg) {
        cout <<"### ERROR ### "<< err_msg << endl;
    }
      toyBox[1].setContainsBattery(true);

    Box<Box<Toy> > multipleBoxes(1,1,1,20);
    multipleBoxes.add(toyBox);
    multipleBoxes.add(toyBox2);
```

```
    cout<<endl<< multipleBoxes <<endl;

    //Book Constructor: label, weight, length, width
    Book book("Terry Pratchett: The Colour of Magic",0.2,0.4,0.2);
    Box<Book> bookBox(0.05,0.2,0.4,2);
    bookBox.add(book);

    cout<<bookBox<<endl;

}
```

Sample Output:

```
Toy Label: Lego: Heroes Batman # 0.2x0.3 1.2kg No Battery

### ERROR ### Index out of bounds!

***** Box<Toy> *****
Box item count:2
Size: 1x1 Box Weight:0.1kg Total/Maximum Allowed Weight:3kg/7kg
Items:
1:Toy Label: Lego: Heroes Batman # 0.2x0.3 1.2kg No Battery
2:Toy Label: Lego: Star Wars Death Star # 0.5x0.3 1.7kg No Battery
********************

### ERROR ### The dimensions of the contained object should be equal or smaller than
those of the box!
### ERROR ### The total weight of the contained objects including the box may not
exceed the maximum allowed weight for the box!

***** Box<Box<Toy>> *****
Box item count:2
Size: 1x1 Box Weight:1kg Total/Maximum Allowed Weight:10kg/20kg
Items:
1:***** Box<Toy> *****
Box item count:2
Size: 1x1 Box Weight:0.1kg Total/Maximum Allowed Weight:3kg/7kg
Items:
1:Toy Label: Lego: Heroes Batman # 0.2x0.3 1.2kg No Battery
2:Toy Label: Lego: Star Wars Death Star # 0.5x0.3 1.7kg Contains Battery
********************
2:***** Box<Toy> *****
Box item count:4
Size: 1x1 Box Weight:0.1kg Total/Maximum Allowed Weight:6kg/7kg
Items:
1:Toy Label: Lego: Heroes Batman # 0.2x0.3 1.2kg No Battery
2:Toy Label: Lego: Star Wars Death Star # 0.5x0.3 1.7kg No Battery
3:Toy Label: Hot Wheels: Turbine Twister Track Set # 0.3x0.3 1.4kg No Battery
4:Toy Label: Lego: Heroes Superman # 0.3x0.3 1.6kg No Battery
********************
********************

***** Box<Book> *****
Box item count:1
Size: 0.2x0.4 Box Weight:0.05kg Total/Maximum Allowed Weight:0.25kg/2kg
Items:
1:Book Label: Terry Pratchett: The Colour of Magic # 0.4x0.2 0.2kg
********************
```

Note: If you face a problem with the assignment or given test program, contact Res. Asst. Çağatay KOÇ as soon as possible via e-mail (kocca@itu.edu.tr) or in person (Res. Lab. 3).

Submission Procedure:

1. Your source code should be named “box.cpp”, “box.h”, “toy.cpp”, “toy.h”, “book.cpp” and “book.h”.
2. Make sure you write your name and number on all the header files of your project with the following format.

```
/*
* @Author
* Student Name: !! enter here !!
* Student ID : !! enter here !!
* Date:
*/
```

3. Make sure that GNU C++ compiler (g++) compiles your project, and the application runs in Linux smoothly. You can use ITU ssh server to compile and test your application. This is important because we will evaluate your homework in Unix using g++.
4. Use comments wherever necessary in your code to explain what you did.
5. After you make sure that everything is compiled smoothly, archive all files into a zip file. Submit this file through www.ninova.itu.edu.tr. Ninova enables you to change your submission before the submission deadline.

Academic dishonesty including but not limited to cheating, plagiarism, collaboration is unacceptable and subject to disciplinary actions.