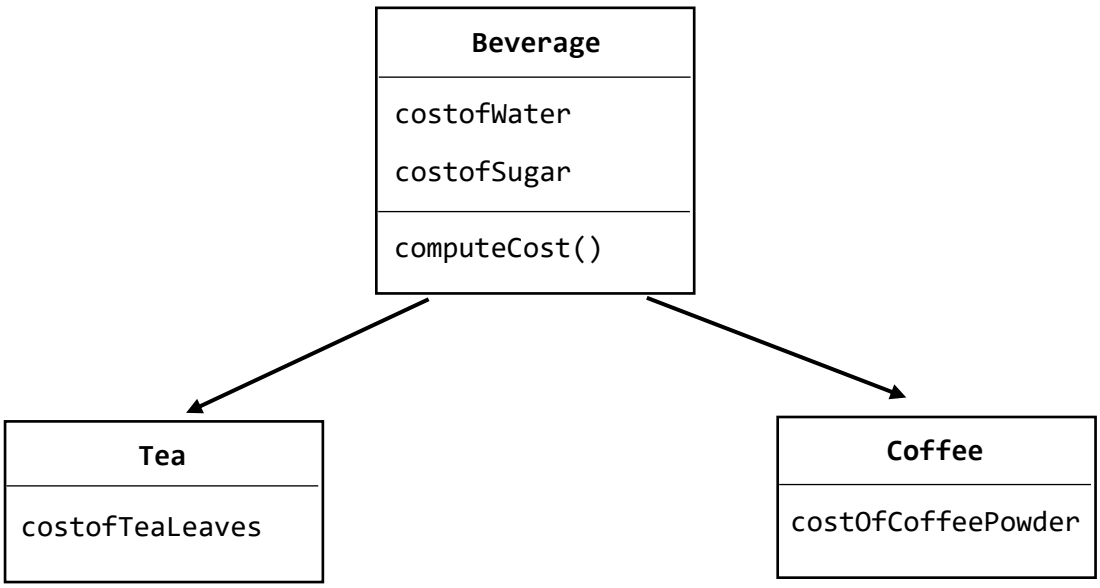
Programming problems

This section contains some programming excercises that will help you understand the basics of all the concepts we have read so far in this course. No need to repeat the question. Just include the following format : Question number, Source Code, Output and Description.

1) Create a class called ***time*** that has seprate int member data for ***hours***, ***minutes*** and ***seconds***. One constructor should initialize this data to 0, and another should initialiize it to fixed values. Another member function should display it, in HH:MM:SS format. The final member function should add two objects of type time passed as arguments. Write a main() function to test your program.

2) Write a C++ program that illustrates how strings can be manipulated via operator overloading.

3) Imagine you are writing an application in C++ for a cafeteria management system. Create a base class called ***Beverage*** inside which include data members to store ***cost of water*** and ***cost of sugar***. Also, create a member function inside ***Beverage*** to ***compute cost*** of a beverage. Create two more classes, call them ***Tea*** and ***Coffee***, derived from class ***Beverage***. Class ***Tea*** needs an extra data member to store ***price of tea leaves***. Class ***Coffee*** does not need this data member but needs another data member to store ***cost of coffee powder***. Create appropriate constructors in respective classes to initialize cost of each ingredient for a beverage from user given values. We are not sure what we need to code inside ***computeCost***() function of class ***Beverage***. However, computing cost of tea and coffee is very simple: Just add the cost of individual ingredients for the respective drink. For example: cost of tea equals cost of water plus cost of sugar plus cost of tea leaves. Finally, write main function to test your program.



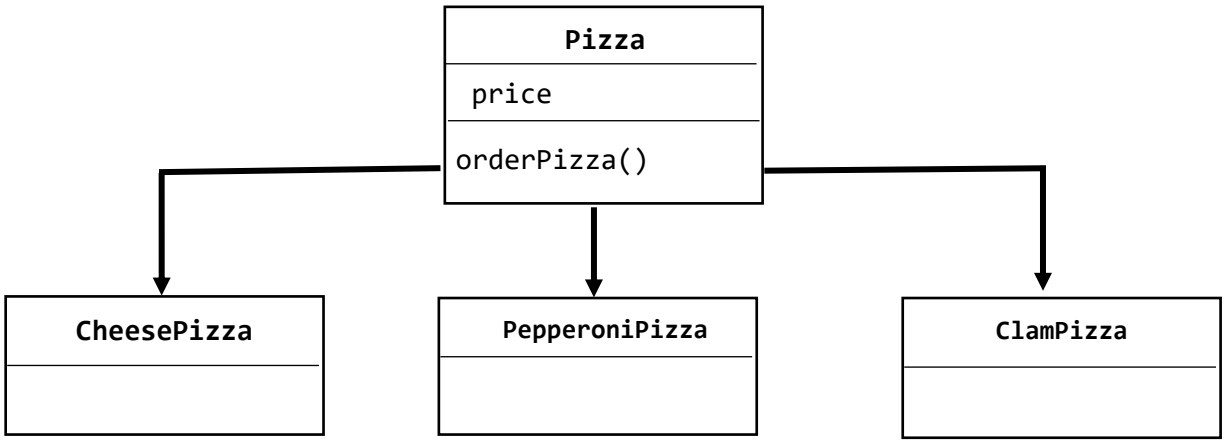
4) Suppose you are developing a C++ application for Domino's Pizza. Create a base class called ***Pizza***. Include data member ***price*** (integer) to store price of a particular pizza and a member function ***orderPizza***() in this class. Now create three more classes called ***CheesePizza***, ***PepperoniPizza*** and ***ClamPizza*** that are derived from class ***Pizza***. You are not sure what you would code inside ***orderPizza***() method of base class ***Pizza***. ***orderPizza***() methods of derived classes should just display to the screen what kind of pizza your customer has ordered and the price of that pizza. Write appropriate constructors inside all the four classes to initialize price of objects created. Inside main function, write code to display menu and wait for user input. After user inputs his/her option, your program should dynamically (use pointers) to allocate memory and order that pizza. Your program output should look like this:

Welcome to Domino's Pizza. Enter your choice.

1. *Cheese Pizza*
2. *Pepperoni Pizza*
3. *Clam Pizza*

Enter your choice: 2 ⏎

You have ordered Pepperoni Pizza. Price: 140



5) Write a C++ program that to illustrate exceptional handling mechanism. Try calculating factorial of a user given number. If the number is negative, factorial cannot be calculated.

6) Imagine writing a C++ application for a video streaming website. Write a class to represent a ***video***. The class should contain four data members to represent ***title*** of the video (string), ***number of views***, ***likes*** and ***dislikes*** (integers). Create parametrized constructor to initialize the data members for the object created. Overload greater than operator that checks if a video is more loved than another video by using friend function (returns bool). A video can be considered

more loved than another if and only if it has more views and likes but lesser dislikes. Write main() function to test your program.

7) Create a base class called ***user*** to represent a user in an application. Include data members ***name*** (string) and ***password*** (string) in class user. Derive two classes from user to represent two different kinds of user : ***admin*** and ***normal*** user. Class admin needs a new data member to store ***phonenumber*** (integer) of administrator. Include a member function called ***authenticate***() in class user. Keep this function pure virtual in base class but override it to perform specific authentication procedure for both derived classes. Define appropriate constructors in all these classes to initialize data members of created objects such that derived class constructor will call base class's constructor. Authenticating normal user is simply matching username and password of an object from user given values. Authenticating admin means matching username, password and phone number of the object from user given values. Inside authenticate() method of derived classes, ask for user's necessary credentials and after necessary comparisons, display appropriate messages. Inside ***main***() function, write code to display a menu to select how your user would like to authenticate. After user enters his/her option, your program should dynamically (use pointers) allocate the memory and ask your user necessary credentials.

Welcome to the application. Who are you?

1.Admin

2.Normal user

1. ⏎ Please enter your username and password.

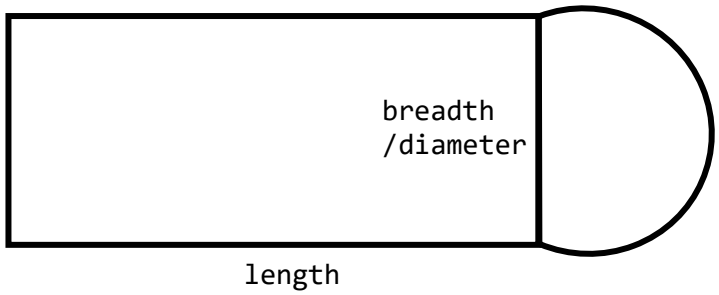
ricky ⏎ pass123 ⏎

Sorry. Wrong credentials.

8) In this program, you will be implementing function template with multiple parameters. Create a function template ***max***() that accepts two arguments of type T1, T2 (T1 and T2 being templates) and returns maximum between these two numbers. Write main() function to test your program.

9) Create a base class called ***shape***. Class shape has one data member call it, ***side1,*** that stores length of one side of a shape***,*** only one function to calculate area, call it ***findArea()*** which is pure virtual. Two classes, call them ***rectangle*** and ***circle***, are derived from shape and override

findArea() according to their use. Class rectangle contains an additional data member to store ***length of another side*** of the rectangle. Class circle do not need this data member, as variable side1 will act as diameter for this class. Define appropriate constructors inside all three classes such that constructors of derived class will call constructors of base class. In this program, you will be writing code inside main function to calculate a portion of a basketball court as shown here. Ask as input two variables from user to store length and breadth (or diameter) and pass these input as arguments to constructor.



Area of rectangle = length x breadth
Area of a half circle = 1/2 x πr^2
radius = half of diameter

10) Create a base class called ***appliance***. Derive two classes from this class, call them ***printer*** and ***scanner.*** Class appliance has two basic functions defined: ***on***() and ***off***(). Class printer has an extra data member to store ***copies to print*** and a member function called ***print***() which prints a string that much times (use for loop) whereas class scanner has a data member to store ***copies to scan*** and an member function called ***scan***() that prints a string that much times (use for loop) . Create a new class called ***ComboDevice*** such that it has properties of both printer and scanner. Create appropriate constructors in this class and other classes (if necessary) to initialize necessary data members. Inside ***main***() function, create object of class ComboDevice. Call member functions of object created from ComboDevice in this order : Turn on the appliance. Print paper, Scan, Turn off the appliance. Print appropriate strings inside each member function. Your program output should look like this :

Appliance turned on.
Printing Printing Printing

Scanning Scanning

Appliance turned off.