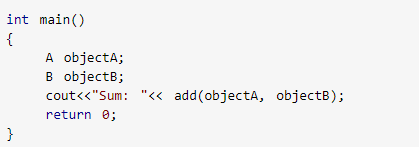
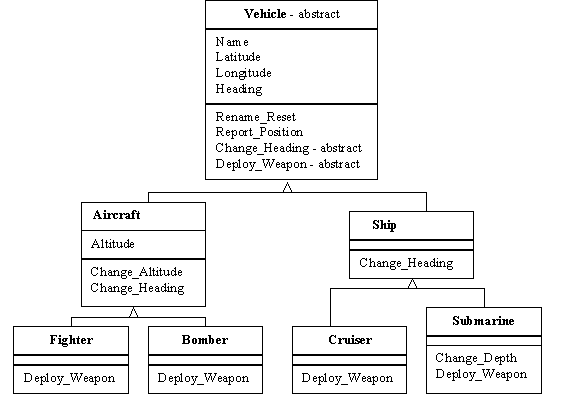
**Practice Questions.**

1. Write a declaration (not a definition) for a const void function called aFunc() that takes one const argument called jerry of type float.
2. What is difference between shallow copy and deep copy?
3. What do you mean by This pointer? Why is it required explain it with an example.
4. Write classes and a friend function for adding two private data members.



1. Create a class student having all essential attributes to store the record of a student (student\_id, semester, course\_registered, marks, CGPA etc.). The class should maintain a record of total number of students enrolled. The registration no. of a student, once assigned, cannot be changed. (Assigned on the time of registration (object creation)).
2. Imagine the publishing company markets both book and audiocassette versions of its works. Create a class called **publication** that stores the title (a string) and price (type float) of a publication. From this class derive two classes: **book**, which adds a page count (type int); and **tape**, which adds a playing time in minutes (type float). Each of the three classes should have a getdata() function to get its data from the user at the keyboard, and a showdata() function to display the data. Write a main() program that creates an array of pointers to publication. In a loop, ask the user for data about a particular book or tape, and use new to create an object of type book or tape to hold the data. Put the pointer to the object in the array. When the user has finished entering the data for all books and tapes, display the resulting data for all the books and tapes entered, using a for loop and a single statement such as pubarr[j]->showdata(); to display the data from each object in the array.
3. Implement the following



1. Create a class **BankAccount** having following attributes:
2. accountHolderName
3. accountNo
4. Balance

And following functions

1. withdraw() pure virtual
2. deposit() pure virtual
3. checkBalance()

Inherit 3 classes implementing the above interface.

**SavingAccount**, **CheckingAccount**, **InvestmentAccount**.

**SavingAccount** generates a profit of 5% on each deposit and deducts an amount of 10% if the customer withdraws amount in the last week of the month.

**CheckingAccount** doesn’t involve any deduction or profit.

**InvestmentAccount** generates 10% profit on each deposit but there is no deduction rule defined so far.

Test your code by generating different bank accounts.

Note: (Abstract class can also be called as an interface and the derived classes providing the implementation of the pure virtual functions are known to be implementing that interface.)

1. Create a class **Employee** having following attributes and functions
2. Name
3. hireYear
4. monthlyPay() pure virtual
5. annualPay() pure virtual

Inherit two classes from Employee class having mentioned attributes

**SalariedEmployee** **HourlyEmployee**

1. monthlySalary i. hoursPerWeek ii. hourlyWage

Create two classes **Manager** and **Staff** implementing the **SalariedEmployee class.**

Define two classes **FullTimeEmployee** and **PartTimeEmployee** implementing the **HourlyEmployee** class.

1. Write a template class **Calculator** having following template functions
2. Add()
3. Subtract()
4. Multiply()
5. Divide()
6. Modulus()

Test your code using different type inputs.

Note: Handle exceptions for string or character inputs.

1. Implement **Stack** using Templates. Create a template class Stack and add following functions
2. Push()
3. Pop()
4. isEmpty()

Test your code with different data type inputs.

Note: You can implement using Arrays or Pointers, that’s up to you. Add proper comments explaining your code to get marks.

1. Create an abstract class Shape having functions Area and Perimeter. Inherit three classes triangle, square and rectangle implementing the above class shape. Write a menu driven code to calculate the area and perimeter of the shape, the user inputs for. Use dynamic polymorphism.