Agenda

- Revision
- Shallow copy and deep copy
- Copy constructor
- Operator overloading
- Conversion function

Shallow Copy, Deep Copy, Copy Ctor (demo01 to demo03)

- When we initialize the new object with an previously created object then copy ctor gets called.
- If we havent provided a copy ctor then the default copy ctor gets called.
- default copy ctor does the shallow copy.
- It only copies the values as it is from one object to the other.
- If your class consists of pointer type of data member and dynamic memory allocation is done then in this case your shallow copy will copy the existing address from one object into the another.
- So any changes you do in these dynamic memory will have effect on both the objects.
- It means your both the objects are pointing at the same dynamic memory allocated.
- so to allocate a seperate memory for the new object and then we have to perform deep copy.
- to perform deep copy we have to provide our own copy ctor.
- copy ctor is a single parameterized ctor which takes the parameter type same as that of the class as reference.

Operator Overloading (demo04 to demo10)

- We cannot use the operators for user defined type of data.
- to allow the operators to work for use defined types we need to overload the operators.
- We can overload the operators as member as well as non member function of the class.

Conversion Function (demo11)

- Conversion Functions are the functions that are used to convert object of one type into another type.
- We can alos use the conversion functions to convert user defined tyeps of object into fundamental type.

Algorithm

- Q2
    - Product class -> id, title,price,type='B' or type 'T'
    - book -> author
    - Tape -> artist
    - Product *p = new Book() or new Tape() {this->type = 'B' or 'T'}
    - calculateBill(productarr[3]) for() if(type = 'B') for() if(tyep = 'T')
- Q3.

- Employee -> string designation = "manager" OR "salesman" OR "salesmanger"

- 1. Accept Employee
    - a. Accept manager -> new manager() -> {this->designation = "manager"}
    - b. Accept SalesMan -> new salesman() -> {this->designation = "salesman"}
    - c. Accept SalesManager -> new SalesManager() -> {this->designation = "SalesManager"}
- 2. Display the count of all employees with respect to designation
    - expected output
    - manger - 3
    - salesman - 4
    - salesmanger - 3
- 3. Display respective designation employees
    - manger
    - salesman
    - salesmanager for(){ if(designation = "") }

- display all employees

Lab work

- Complete assignment 7
- Go through the classwork if required
- complete the assignment 8
- implement the copy ctor and assignment operator overloading inside the matrix assignment as per the requirement. (optional)