OOP

Object oriented Programing is a method used for designing a program using classes and objects.

Object oriented programming:

) Procedural programming:
Writing functions or procedures that perform operations on data

) Object oriented programming is about creating objects that contain both data and objects.

Advantages of OOP

1) Allow to break program into the bit-sized problems that can be solved easily.
2) To keep the code DRY "Don't Repeat Yourself". debug, modify
3) Makes code easier to maintain. h
4) Provide clear structure for program
5) Make it possible to create full reusable applications with less code and shorter development time.
6) Faster and easier to execute
7) Using OOP we can achieve
   i) Abstraction
   ii) Encapsulation
   iii) Polymorphism
   iv) Inheritance.

have well defined attributes and behavi

Class

1) User defined data type that we can use in our program,
2) works as an object constructor or a blueprint for creating objects
3) Class has its own data members and member functions.

Object :

1) object is a real world entity
2) self contained components, which consists of methods and properties to make a particular data useful.
3) Instance of a class with its own data members and member functions.

class	object
logical entity	2) physical entity.
	3) real world entity
Blueprint of object used to declare and create objects	

Objects logic are done by classes.
By Phone we can
i) call
ii) Bluetooth
iii) take photo.
those every logic will be divide as classes.

Disadvantages of OOP:

1) Size —
Object oriented programs are much larger ~~and~~ than other programs.

2) Effort :
OO Programs require a lot of work to create

3) Speed :
OO programs slower than other programs becaus of their size.

---

when we say x language is OO ?
it mean every problem will be solved with the perspective of objc

Abstraction :

main purpose is to hide the unnecessary detail from the users

In English it means
→ only show relevant data ~~of~~ details.
→ rest of others are hide.

→ This is mostly done by interfaces rather than abstract classes.

→ when there are some common features to be shared by all the objects, then we go for abstract class
→ when all the features are to be implemented differently for all the objects, then we go for interfaces.

---

see also java C# note

) abstraction lets you focus on what the object does, instead of how it does it.
→ abstraction is the process of hiding the implementation details and showing only functionality to the user.
) Data Abstraction is the act of representing the essential feature without knowing the background details.

(5) car → we don't know how break system, car start system wor
(example best for both Abstract & Encap

Real world examples of abstraction :

1) ATM machine
2) we use bluetooth but we don't konow how it connect with other phones & devices
3) we use washing machines but we not know how it work

Encapsulation :

A process of wrapping code and data together into a single unit

Encapsulation is :

1) the bundling of data along wi the methods that operate on th data into a single unit.
2) It may also refers to a mechnism of restricting the direct acces to some componen of an object. ~~For example~~

For Example

user cannot access data members directly.

(can be used to hide both data members and functions)

3) A class is an example of encapsulation in that it consi of data and methods that have been bundled into a single unit (object).

In real time we are using encapsul. for security purpose.

1) Encapsulation is the attribute of an object and it contains all data ~~that~~ which is hidden. That hidden data can be restric to the members of that class.

2) Levels are public, protected, private, internal and protected internal.

To achieve encapsulation

1) declare data members of the class private

2) privide get and set methods to access and update the value of private variable. (where we can control using different checks if (data not null) { → }

Real life Examples of Encapsulation :

1) School bag is an example of encapsulation
   → School bag can keep our books and pens etc.

2) Capsule is also example.
   → Capcul encapsulates several combinations of medicine.

3) Suppose you have account in bank. If your balance variable is public in this case ~~your balance is~~ everyone can know your balance.

to achieve data hiding

So we declare balance variable as private and the person who want to access balance is by using the methods defined in the class by giving id, name, password.

Thus we achieve security by utilizing the concept of data hiding. This is called encapsulation

* → Encapsulation is a method to hide the data in a single unit along with the methods to protect information from the outside.

Data Hiding:

1) Outside users can't access our internal data directly or our internal data should not ~~go~~ so out directly. This oops feature is Data Hiding.

After validation & authentication outside user can access internal data.

Example:

gmail account is accessible only after authentication.

...complete implementation

Interface → define "contract" that all classes inheriting from should follow

Inheritance

1) Inheriting the properties of parent class into a child class is called Inheritance.
2) Inheritance represent is a relationship (parent - child).

See also in (OOP 1, Pg 177)
1) Association (no intrinsic rel)
2) Composition (death relationship)
3) Aggregation.

~~AoB~~
{ ~~AXoB~~ }

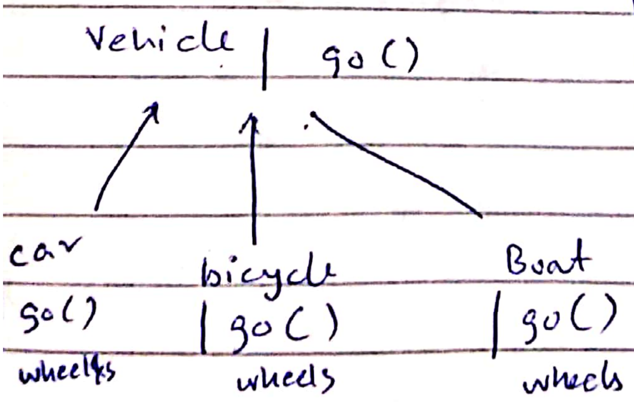
~~Asso~~

Association — Ali —lives in→ House

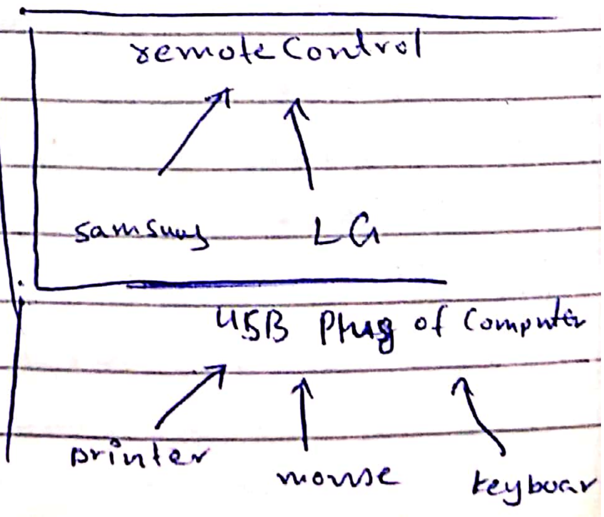
Composition — House ← rooms, ← Doors

Aggegation — Room ← Table, ← chair, ← bed, ← furniture

Abstract class:



Interface:



OOP 1 Pg 214

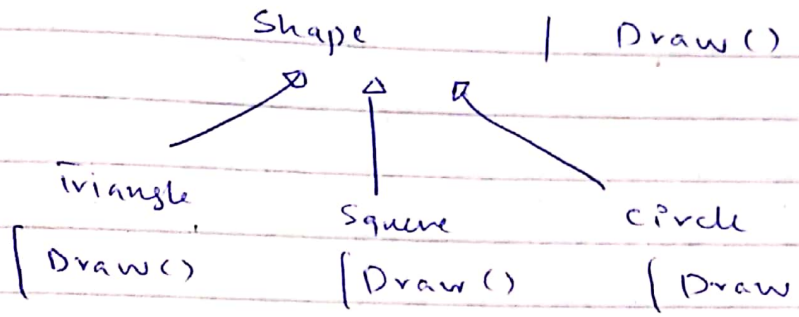
(9)

Polymorphism is the ability of an object to take on many forms

Polymorphism :

Objects of different classes related by inheritance respond differently to the same message.

Example (1)




```
Shape * p = Triangle | Square | Circle
p → draw()
```

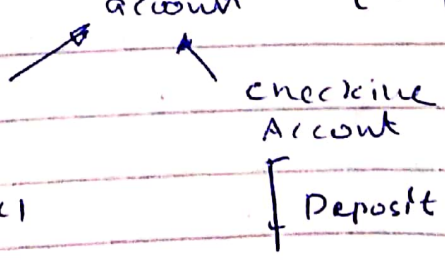
p - Draw behave differently for the same message (p → Draw()), for different objects.

Example (2)

we can turn on or turn off our phone by the same button.

Example (3)

```
Account * p;
p = &SA;
p = &CA;
p → Deposit()
```



Why DSA

Why Data Structures

1) Enable the programmers to handle data effic
2) DS are nessassary for designing efficient algorithms.
2) Using appropriate data structure can help programmers save a good amount of time while performing operations such as
   1) storage
   2) retrieval
   3) processing
   4) Searching
   5) Sorting.

3) Data Structure is a way to store and organize data in order to facilitate access and modification.

Stack :

→ Last In first out (LIFO)
→ Element added first will be removed first.

→ insertion (push)] from top
removal (pop)] of stack

Examples
1) Stack of Books
2) Weighting bar.
3) Undo / Redo Behaviour.