14. Inheritance (OOP)

[Ignore GUI]

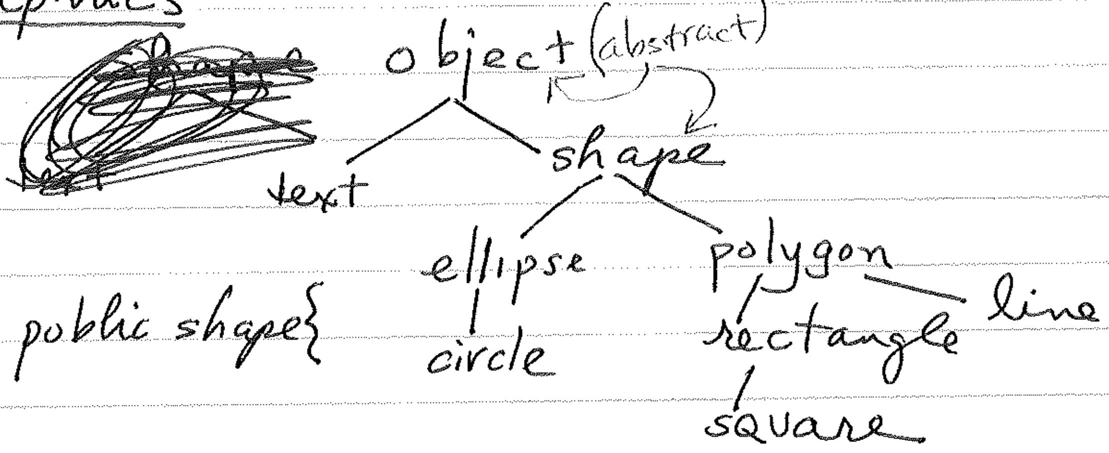
Stroustrup ch 14

1

Polymorphism ~~ad hoc~~

ad hoc	- conversion = coercion
	- overloading
universal	- inclusion, inheritance overriding virtual fns
	- parametric = template = generic

ex: graphics

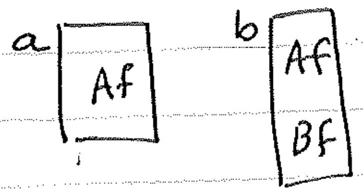


```
class ellipse : public shape{
}
```

- minimum set of opns & fns each class
- prefer non-member when possible
- same name similar opns
- virtual fns when diff in classes
  ⇒ overriding
- any virt fn op ⇒ require virt. dtor

Stroustrup ch14
2

```
class B : A {
} A a; B b;
```



a = b
copies only A fields
∴ disable copyctor &
op = in base class
∴ can only pass by
reference or pointer
⇒ make them private
& don't implement

Protection

public - accessible anywhere
protected - accessible to itself & friends
its derived classes
private - only itself & friends

Ctor 1st - call base class ctor
then: selfctor

Dtor first - delete self
then - call base class dtor

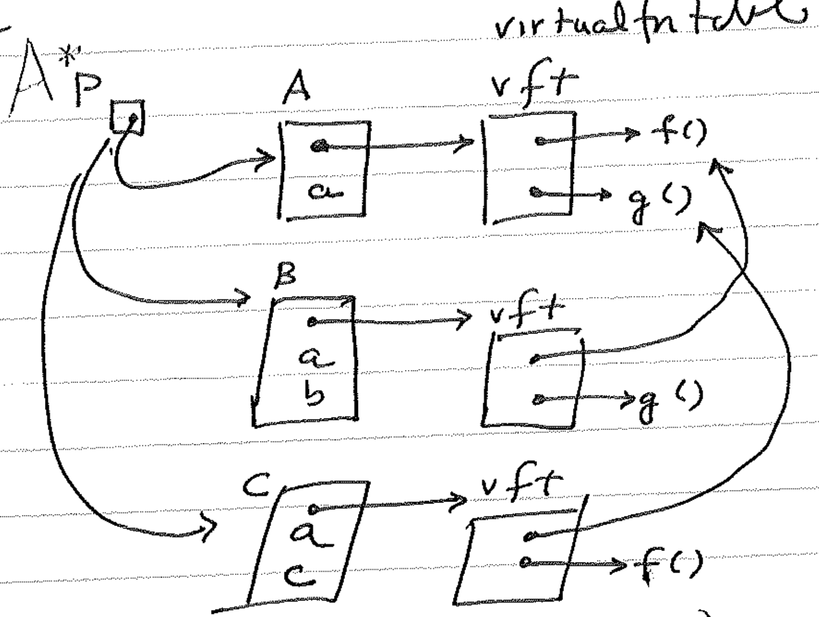
Stroustrup ch 14
3

- overriding - replace virt fn in base class w one in derived

- Use ptrs & refs only, never by value
  - disable copy ctor & op = → make <u>private</u>

<u>abstract</u> base class
- used only as base
- protected ctors prevent creation
- may have abstract virtual fns
  ex: virtual void f()=∅;
  - pure virt fns
- must have virtual dtor if abstract or pure

storage layout
```
class A {
    a
}
class B : A {
    b
}
class C : A {
    c
}
```



<u>sizeof</u> one ptr overhead ∀ base class

$p \to f(x,y)$
$\Downarrow$
$(p \to vft \to f)(p, x,$

Stroustrup 14

4

type info

Java p.getClass().getName()

C++ typeid (*p).name()

Avoid casts

dynamic_cast <T*> (p)

or returns null

14.3 Base & derived classes

- derived: adds fields to base

maybe overrides virtal fns

- virtual fns - runtime polymorphism

$p \rightarrow f(x) \Rrightarrow (p \rightarrow vft \rightarrow f)(p, x)$

- ~~enc~~

Encapsulation: keep details private or protected

Call fn: ptr to vtable to fns addr

pass ptr as left arg

- only virtual fns & virtual dtor in vtable
- can not have virtual ctors
- pass objects by ptr.

Stroustrup 14
5

void f(const object o) ← BAD

void f(const object *p) ← OK
or &p

Override : use same name & type
as in base

~~[illegible]~~

~~publ~~
~~public~~
~~private~~
~~prote~~

14.3.3 Pure virtual fns
virtual void f() =0;
must be overridden in any
non abstract clas,
can not instantiate an abstract class

Java has abstract classes
also: interfaces are abstract

C++ interfaces
: use multiple inheritance
- no fields & all ~~abstr~~ pure virt fns

better: use nested classes
e.g. iterators