5/9/24

class-16 :-

→ Java case study. → how many heard about HAS-A , IS-A

HAS-A ⇒ tells components of a particular class (defines composition)

field → define state

→ When we instantiate an object, use constructor to initialize the state.

Constructor returns reference to the object. (types are left for now)

→ system.out is a field in class System: → capital convention in Java.

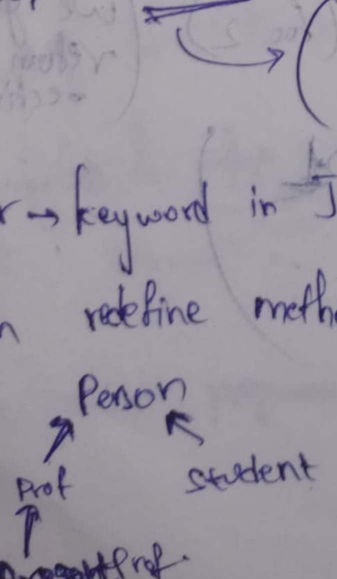
println () → method of object which System.out refers to.

out is a static field, which can be accessed for class & not only for an instance of class.

→ Prof IS-A person (defines inheritence)

super → keyword in JAVA to call constructor of class from which extended.

→ Can redefine methods of parent class.

→ 

← Each of them inherit ______ from parent class.

Javalangobj → global env, no parent
all has (even Person has parent) inherit '

→ The only reason inheritance exists bcz of runtime polymorphism.

→ Person need not be prof,
they can be students.

→ subtypes can replace superset
but not otherway round i.e., if argument is person
I can be prof.

→ Type caste : fools the compiler.

→ polymorphism ⇒ multiple forms
who ~~does~~ polymorphism in OPLs? "Methods"
(Process you are executing)

→ what construct brings Non-determinition in program?

If-Else

```
Person p ;
if (*) random number %
{
    p = new Prof ( );
else {
    p = new stud))
}
p.foo() → If in runtime p is pointed
                to Prof()
          then proff's foo() is called
          If in runtime p is pointed
                to stud()
          then Students foo() is called.
```

→ super class cannot be
type casted to sub-class.