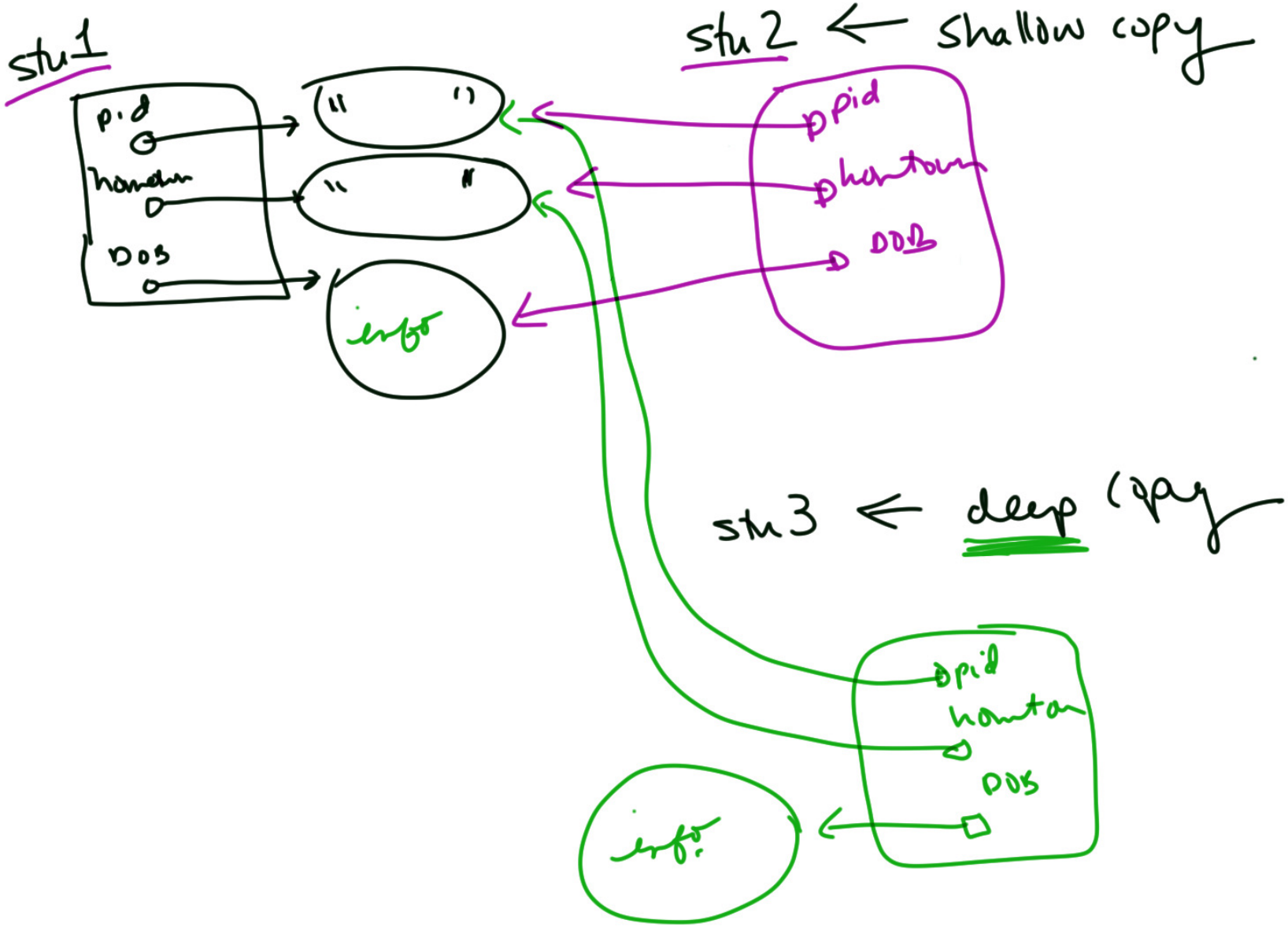
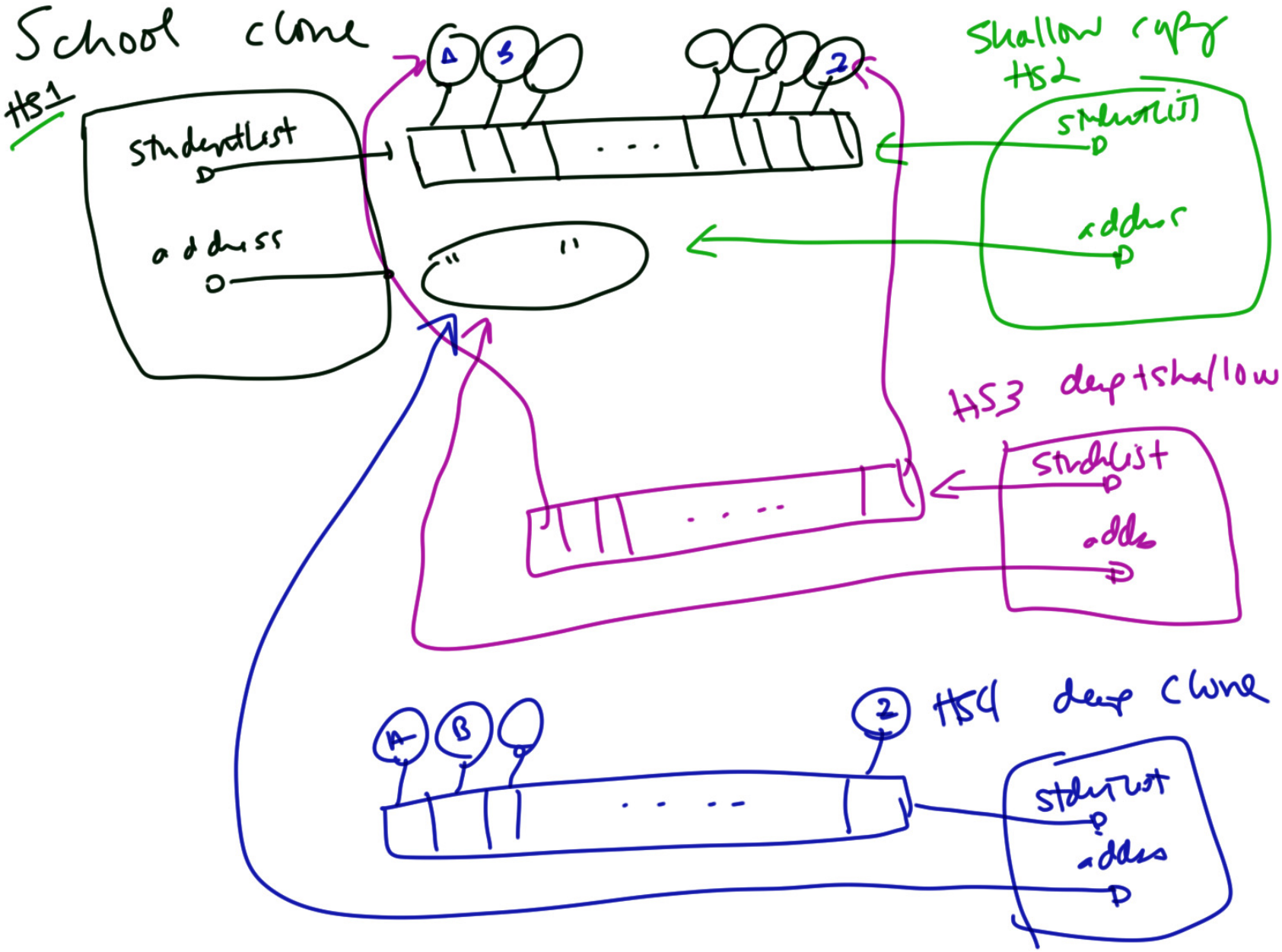
Typical Steps in a public clone method

- Invoke the clone method of the superclass by writing super.clone() (makes a shallow copy)
- Enclose this call to clone in a try block and write a catch block to handle the possible exception CloneNotSupportedException. You can skip this step if super.clone() invokes the public clone method.
- **Clone the mutable data fields of the object that super.clone() returns, when possible.**
- Return the clone

```
153     /**
154      * deep clone
155      */
156     public Object clone()
157     {
158         Student theCopy = null;
159
160         try
161         {
162             theCopy = (Student)super.clone();
163         }
164         catch (CloneNotSupportedException e)
165         {
166             System.err.println("Cannot clone: " + e.toString());
167         }
168
169         theCopy.DOB = (Date) DOB.clone();
170         return theCopy;
171
172     }
```


stu1
stu2 ← shallow copy
p.d
name
DOS
ppid
phantom
DOB
info
stu3 ← deep copy
pid
name
DOS
info

```java
    /**
     * deep clone
     */
    public Object clone() {
        School theCopy = null;

        try {
            theCopy = (School) super.clone();
        } catch (CloneNotSupportedException e) {
            System.err.println("Cannot clone: " + e.toString());
        }

        theCopy.studentList = studentList.clone();

        for (int i = 0; i < studentList.length; i++) {
            theCopy.studentList[i] = (Student)studentList[i].clone();
        }

        return theCopy;

    }
```


School clone
HB1
studentList
address
A
B
2
Shallow copy
HS2
studentList
address
HS3 deep shallow
studList
addr
HS4 deep clone
studList
addr