

# Inheritance & Polymorphism

derived classes

virtual methods

dynamic\_cast operator

# Inheritance (derived classes)

```
class FeupPerson {  
public:  
    FeupPerson(int id, string name, string address);  
    void showRecord();  
    void changeAddress(string newAddress);  
protected:  
    int id;  
    string name;  
    string address;  
};
```

```
class Student : public FeupPerson {  
public:  
    Student(int id, string name, ... , int year);  
    void showRecord(); //redefinition  
    void addCourseTaken(Course *newCourse);  
    ...  
private:  
    string programme;  
    int year;  
    vector<Course *> coursesTaken;  
};
```

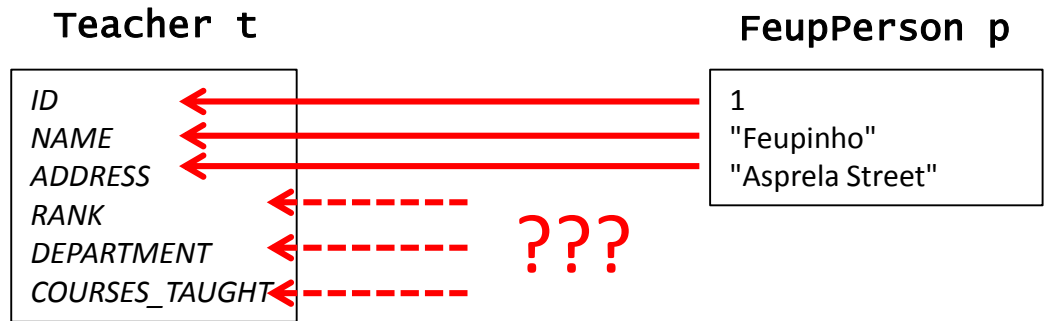
```
class Teacher : public FeupPerson {  
public:  
    Teacher(int id, string name, ...,  
            string department);  
    void showRecord(); //redefinition  
    void addCourseTaught(Course *newCourse);  
    ...  
private:  
    string rank;  
    string department;  
    vector<Course *> coursesTaught;  
};
```

NOTE the type of the coursesTaken and coursesTaught vectors elements

# Slicing problem

- FeupPerson p(1,"Feupinho","Asprela Street");
- Student s(201500007,"Jaime B.,"B. Street", "MIB", 3);
- Teacher t(208785,"Rui Sousa","Sousa Street",..., "DEI");

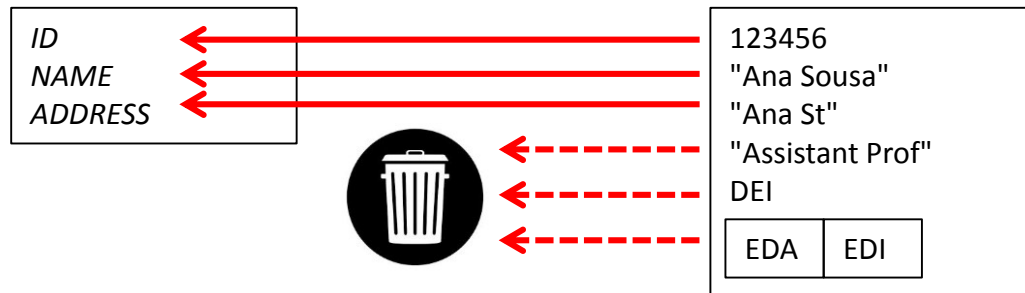
- ...
- ~~t = p; // IMPOSSIBLE~~
- ~~s = p; // IMPOSSIBLE~~
- ...



- p = s; // possible ... BUT SOME DATA IS SLICED AWAY
- p = t; // possible ... BUT SOME DATA IS SLICED AWAY

FeupPerson p;

t = Teacher(123456,"Ana Sousa", ...)



# Slicing problem

- `vector<FeupPerson> p(5000);`
- `p[0] = Teacher(123456, "Ana Sousa", ...);`
- `p[1] = Teacher(123457, "Rui Castro", ...);`
- ...
- `p[4999] = Student(201400500, "Pedro Silva", ...);`

`vector<FeupPerson> p(5000)`

0	ID NAME ADDRESS
1	ID NAME ADDRESS
2	ID NAME ADDRESS
	...
4999	ID NAME ADDRESS

`Teacher(123456, "Ana Sousa", ...)`

123456
"Ana Sousa"
"Ana St"
"Assistant Prof"
DEI
EDA
EDI

`Student(201400500, "Pedro Silva", ...)`

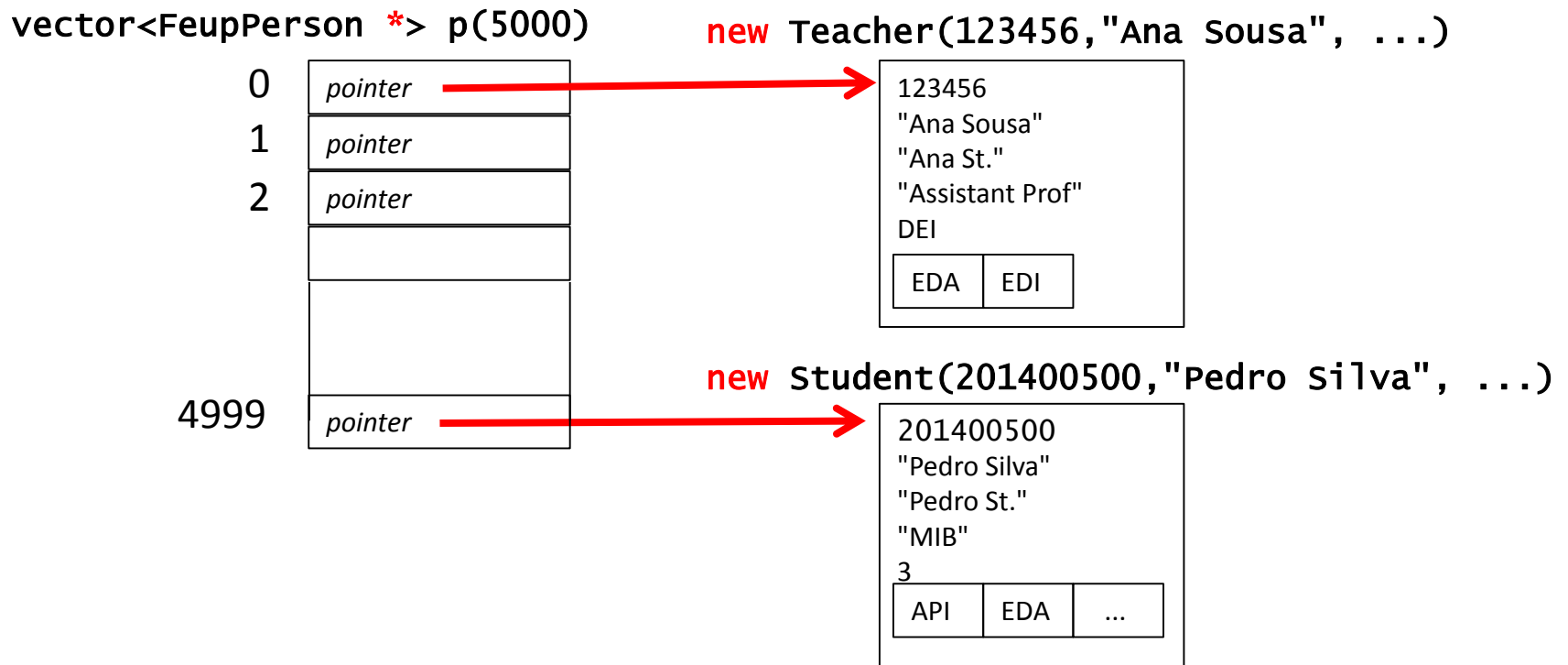
201400500
"Pedro Silva"
"Pedro St"
"MIB"
3
API
EDA
...

?!

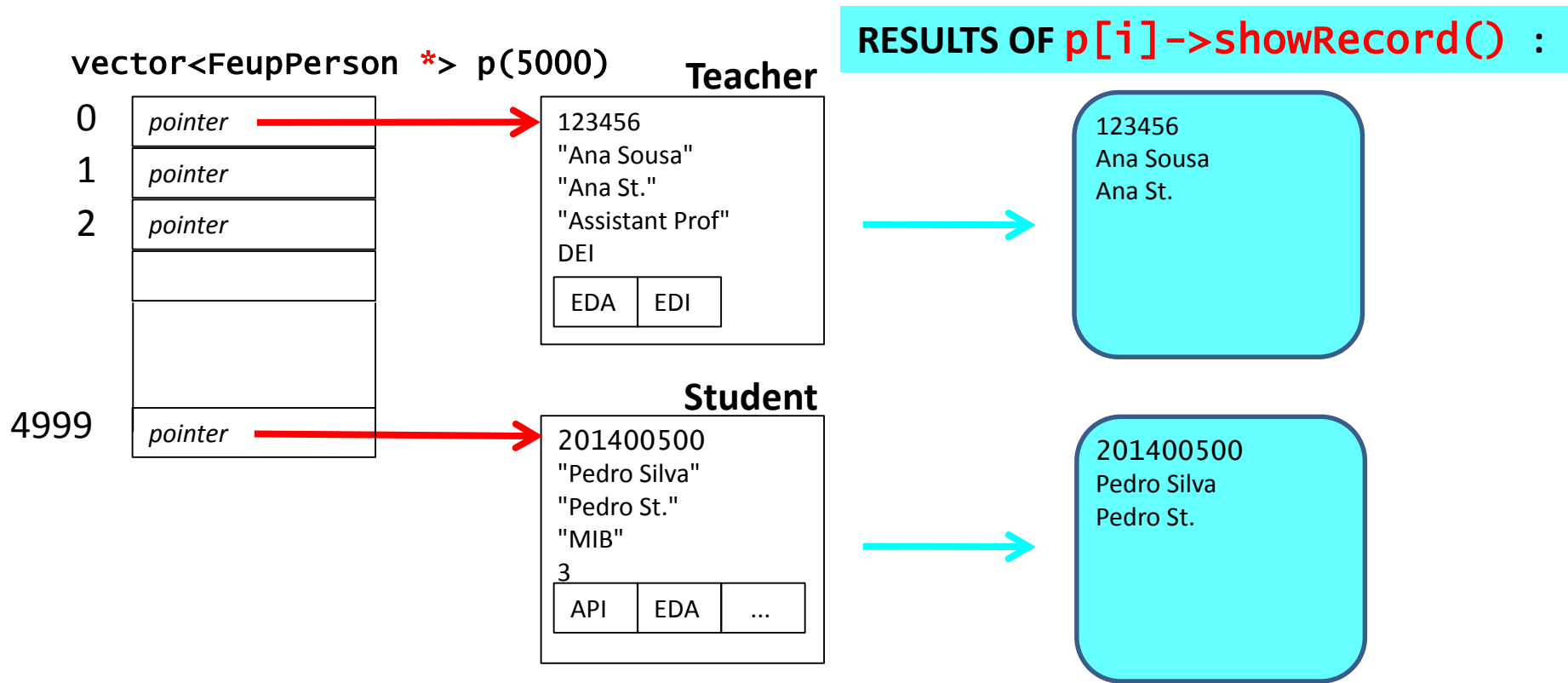
?!

# Solution to slicing problem

- `vector<FeupPerson *> p(5000);`
- `p[0] = new Teacher(123456,"Ana Sousa", ...);`
- `p[1] = new Teacher(123457,"Rui Castro", ...);`
- ...
- `p[4999] = new Student(201400500,"Pedro Silva", ...);`

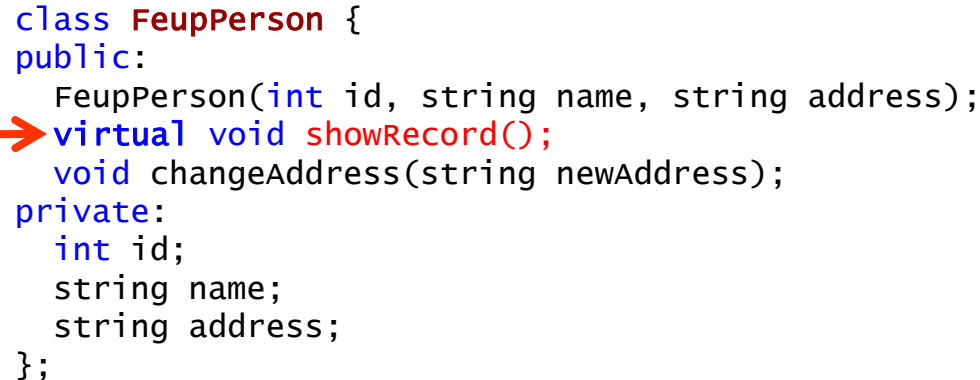


# ... but ...!



# Solution: make showRecord() "virtual"

```
class FeupPerson {  
public:  
    FeupPerson(int id, string name, string address);  
    virtual void showRecord();  
    void changeAddress(string newAddress);  
private:  
    int id;  
    string name;  
    string address;  
};
```

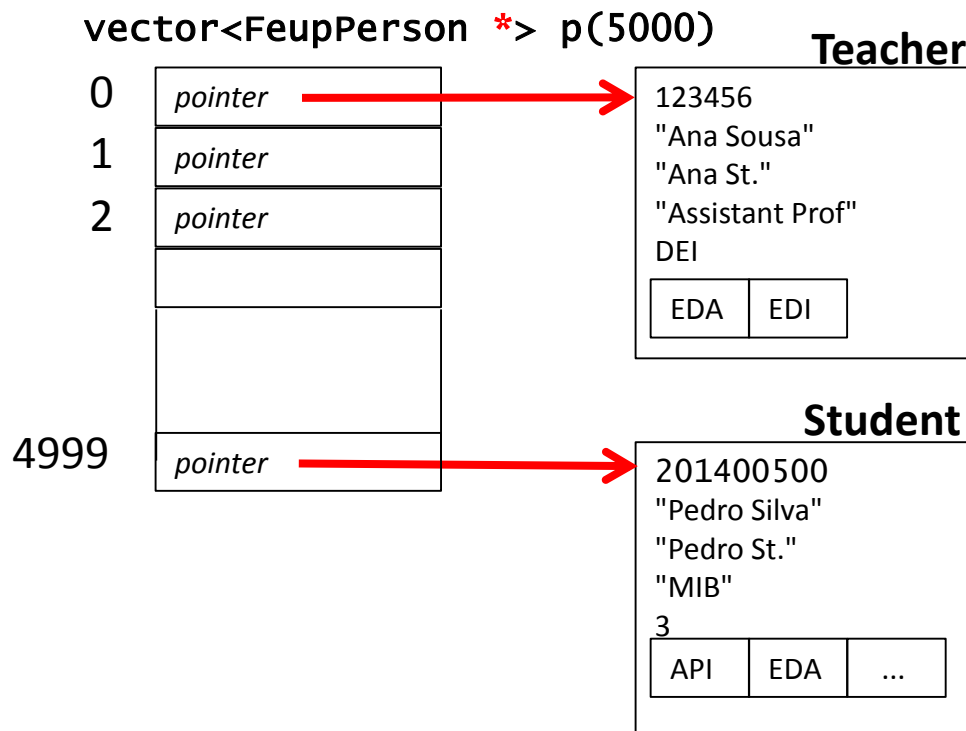


These are  
not mandatory

```
class Student : public FeupPerson {  
public:  
    Student(int id, string name, ... , int year);  
    virtual void showRecord();  
    void addCourseTaken(Course* newCourse);  
    ...  
private:  
    string programme;  
    int year;  
    vector<Course*> coursesTaken;  
};
```

```
class Teacher : public FeupPerson {  
public:  
    Teacher(int id, string name, ...,  
            string department);  
    virtual void showRecord();  
    void addCourseTaught(Course* newCourse);  
    ...  
private:  
    string rank;  
    string department;  
    vector<Course*> coursesTaught;  
};
```

# Solution: make showRecord() "virtual"



RESULTS OF **p[i]->showRecord()** :

123456  
Ana Sousa  
Ana St.  
Assistant Prof  
DEI  
EDA  
EDI

201400500  
Pedro Silva  
Pedro St.  
MIB  
API  
EDA  
...



# ... but ...!

```
vector<FeupPerson *> p(5000);
```

```
p[0]->addCourseTaught(...);
```

OR

```
p[4999]->addCourseTaken(...);
```

**NOT POSSIBLE ...! WHY?**

```
class FeupPerson {  
public:  
    FeupPerson(int id, string name, string address);  
    virtual void showRecord();  
    void changeAddress(string newAddress);  
private:  
    int id;  
    string name;  
    string address;  
};
```

```
class Student : public FeupPerson {  
public:  
    Student(int id, string name, ... , int year);  
    virtual void showRecord();  
    void addCourseTaken(Course* newCourse);  
    ...  
private:  
    string programme;  
    int year;  
    vector<Course*> coursesTaken;  
};
```

```
class Teacher : public FeupPerson {  
public:  
    Teacher(int id, string name, ...,  
            string department);  
    virtual void showRecord();  
    void addCourseTaught(Course* newCourse);  
    ...  
private:  
    string rank;  
    string department;  
    vector<Course*> coursesTaught;  
};
```

# Solution: dynamic\_cast

```
vector<FeupPerson *> p(5000);

Course *c1 = new Course("EDA");
Course *c2 = new Course("EDU");
...

for (unsigned int i=0; i<p.size(); i++)
{
    Teacher *t = dynamic_cast<Teacher *> (p[i]);
    if (t != NULL)
        t->addCourseTaught(c1);
    else
    {
        Student *s = dynamic_cast<Student *> (p[i]);
        if (s != NULL)
            s->addCourseTaken(c2);
    }
}
```