# Polymorphic Lists

**Creating a list (`vector` or array) of different kinds of object also leads to** slicing:

```
vector<Person> v;
v.push_back(Student("Sam", 201795));     // OOPS!!!
v.push_back(Person("Pam B."));
```

When you **`push_back`** a **`Student`** or **`Employee`** object, **the object is sliced** when it is copied into the **`vector`**. The **`vector v`** **does not** contain a **`Student`** and a **`Person`**; it contains two **`Person`** objects. Sam has been stripped of everything that makes him a **`Student`**; he has been **effectively lobotomized**; he no longer knows who he is.

You also cannot fall back on using references, like you did with polymorphic functions, since you cannot create a **`vector<Person&> v`** or an array, **`Person& a[3]`**. **Both of these declarations are illegal.** A reference is not a variable or object (***lvalue***), but an **alias** for an existing ***lvalue***.