STRUCTURES AND POINTERS

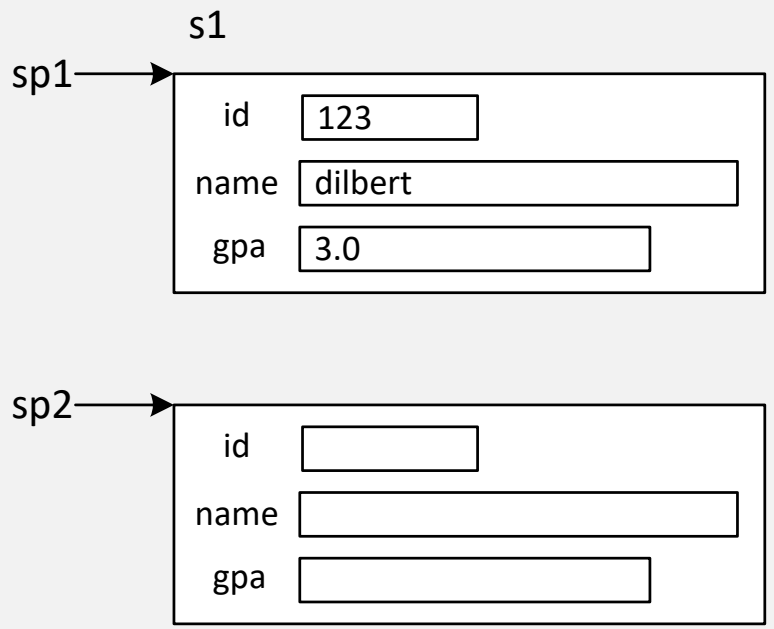
Pointers To Structures

Delroy A. Brinkerhoff

POINTING TO A STRUCTURE

```
struct student
{
    int    id;
    string name;
    double gpa;
};

                . . . .
student  s1 =
       { 123, "dilbert", 3.0 };
student* sp1 = &s1;
student* sp2 = new student;
```



MEMBER SELECTION REVISITED

```
cout << s1.id << endl;
cout << sp1->name << endl;
cin >> sp2->gpa;
```

- `s1.id`
- `sp1->name`

MEMBER SELECTION REVISITED

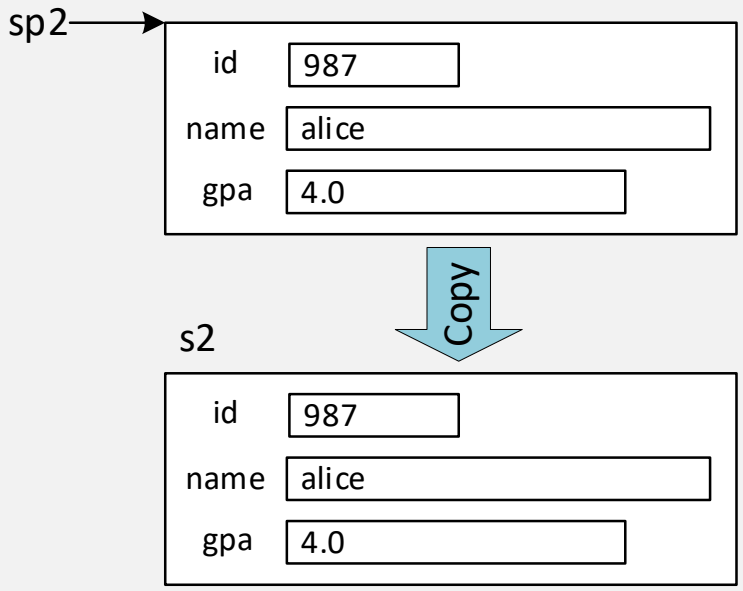
```
cout << s1.id << endl;
cout << sp1->name << endl;
cin >> sp2->gpa;
```

- `s1.id`
- `sp1->name`

INDIRECTION / DEREFERENCING

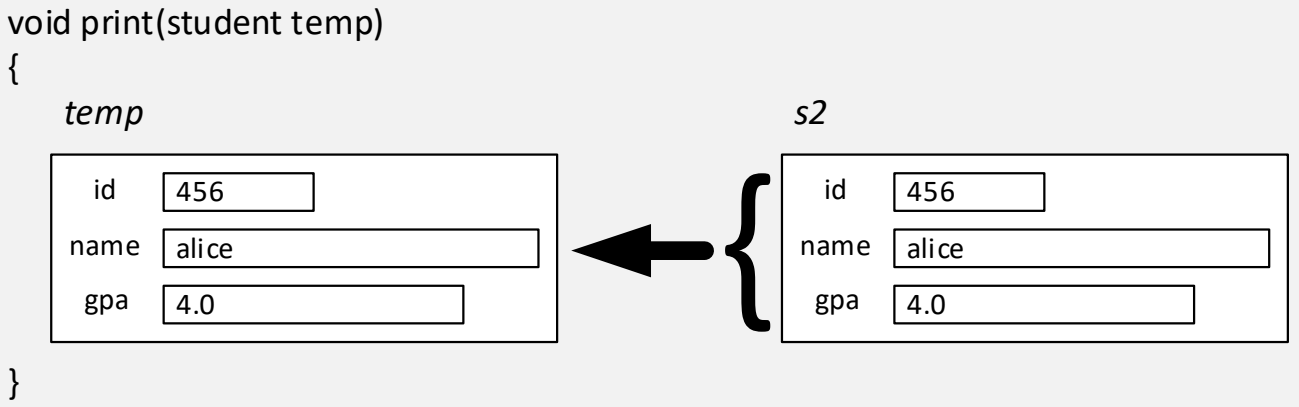
```
student* sp2 =
       new student { 987, "alice", 4.0 };


student  s2 = *sp2;
```



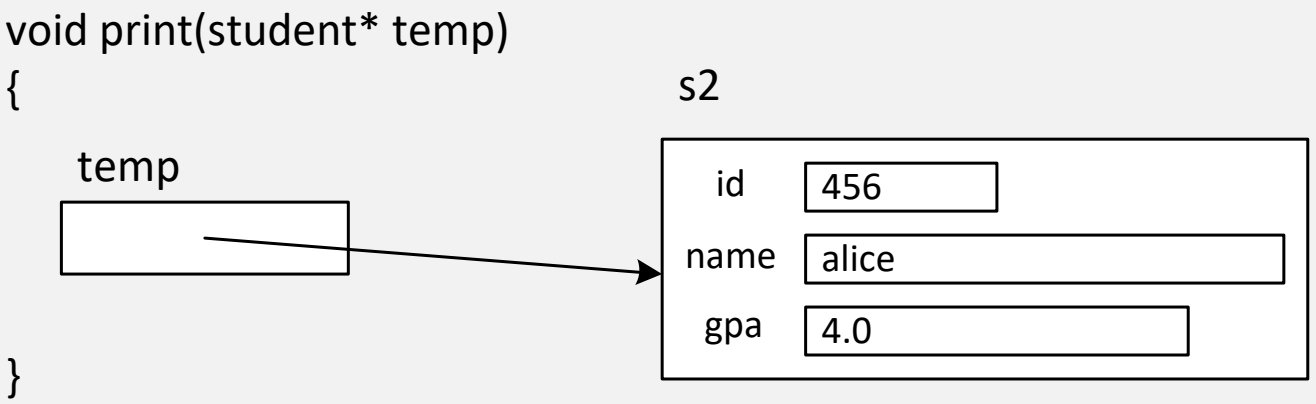
STRUCTURES AS FUNCTION ARGUMENTS

```
void print(student temp)
{
    cout << "ID:   " << temp.id << endl;
    cout << "Name: " << temp.name << endl;
    cout << "GPA:  " << temp.gpa << endl;
}
        .
        .
        .
print(s2);
```



POINTERS AS FUNCTION ARGUMENTS

```
void print(student* temp)
{
    cout << "ID:    " << temp->id << endl;
    cout << "Name:  " << temp->name << endl;
    cout << "GPA:   " << temp->gpa << endl;
}
        .
        .
        .
print(&s2);
student* s3 = new student;
print(s3);
```



IN AND OUT ARGUMENTS

```
void read(student* temp)
{
    cout << "Enter a student id: ";
    cin >> temp->id >> endl;
    cout << "Enter a student name: ";
    cin >> temp->name >> endl;
    cout << "Enter a student gpa: ";
    cin >> temp->gpa >> endl;
}
        .          .          .          .
student   s;

read(&s);
```