Announcements

MP2 available, due 9/15, 11:59p. EC: 9/8, 11:59p.

Arrays: static (stackic)

```
int x[5];
```

Stack memory

loc	name	type	value

Arrays: dynamic (heap)

```
int * x;


int size = 3;

x = new int[size];


for(int i=0, i<size, i++)

    x[i] = i + 3;


delete [] x;
```

Stack memory

loc	name	value

Heap memory

loc	name	value

A point to ponder: How is my `garden` implemented?

```
class garden{
public:
…
// all the public members
…
private:
   flower ** plot;
   // other stuff
};
```

Option 1:

Option 2:

Option 3:

Option 4:

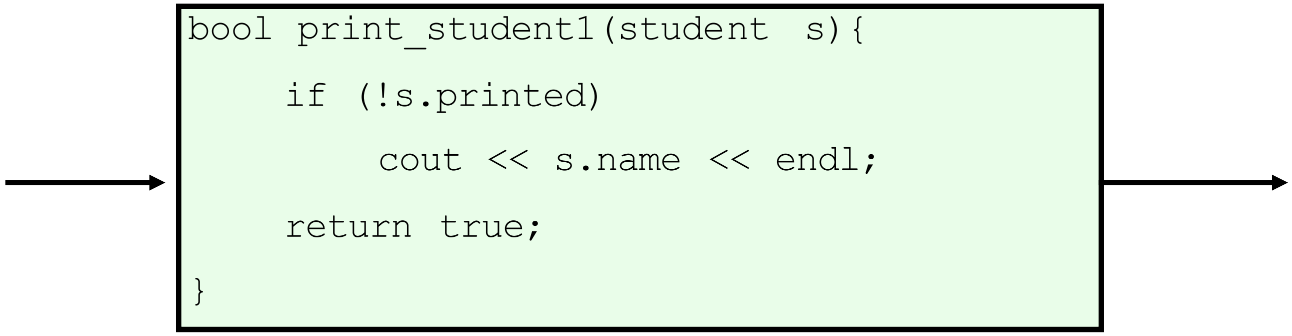
Parameter passing:

```
struct student {
    string name;
    PNG mug;
    bool printed; // print flag
};
```

What happens when we run code like this:

```
int main() {
    student a;
    print_student1(a);
}
```

?



```
bool print_student1(student s){
    if (!s.printed)
        cout << s.name << endl;
    return true;
}
```

Parameter passing:

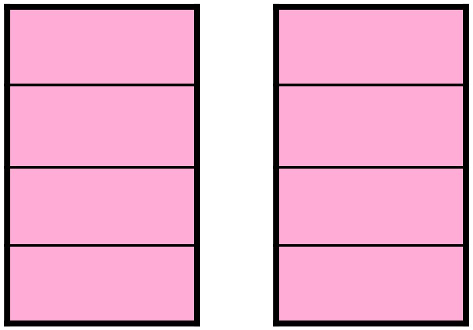
```
struct student {
    string name;
    PNG mug;
    bool printed; // print flag
};
```

Function defn

```
bool print_student1(student s){
    if (!s.printed)
        cout << s.name << endl;
    return true;
}
```

Example of use

```
student a;
… // initialize a
a.printed = print_student1(a);
cout << a.printed << endl;
```



Parameter passing:

```
struct student {
    string name;
    PNG mug;
    bool printed; // print flag
};
```

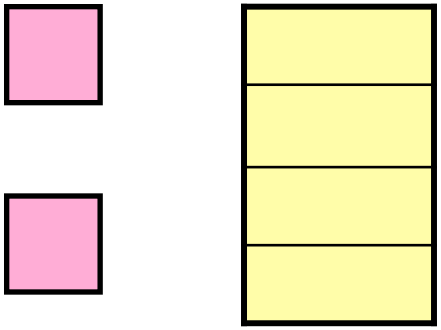
Function defn

```
void print_student2(student     s){
    if (!  s.printed)
        cout <<   s.name << endl;

}
```

Example of use

```
student * b;
… // initialize b
print_student2(b);
cout << b.printed << endl;
```



```
struct  student  {
     string  name;
     PNG  mug;
     bool printed;  // print  flag
};
```

Parameter passing:

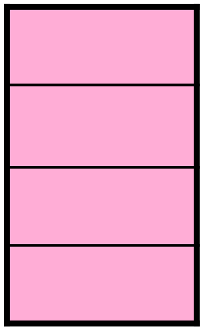
Function defn

```
void print_student3(student     s){
     if  (!   s.printed)
          cout  <<    s.name  <<  endl;

}
```

Example of use

```
student  c;
… // initialize  c
print_student3(c);
cout  <<  c.printed  <<  endl;
```



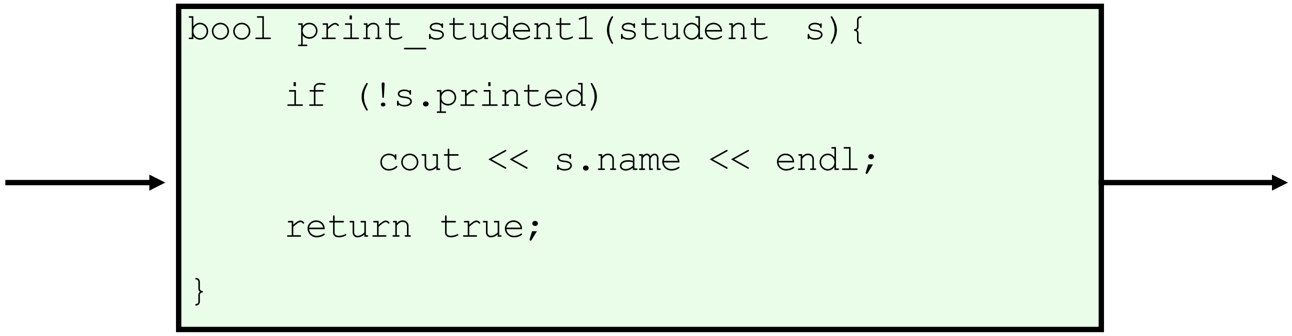
Return values:

```
struct student {
    string name;
    PNG mug;
    bool printed; // print flag
};
```

What happens when we run code like this:

```
int main() {
    student a;
    bool b = print_student1(a);
}
```

?



```
bool print_student1(student s){
    if (!s.printed)
        cout << s.name << endl;
    return true;
}
```

Return by ____________ or ______________ or _______________ .