Today's announcements

- MP2, due on 02/09, 11:59pm

- lab_memory_gdb due on Tue, Feb 02/09, 11:59pm

- First in-lab exam, Feb 10, 11, 12!

  Make sure to come to the lab to which you are registered, only then you will be graded!

Parameter passing:

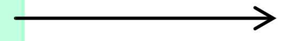
```
struct student {
   string name;
   PNG face;
   int age;
};
```

What happens when we run code like this:

```
int main() {
   student b;
   int new_age = birthday(b);
   return 0;
}
```

?

```
int birthday(student s) {
   s.age++;
   cout << "A birthday!!! " << s.age << endl;
   return s.age;
}
```

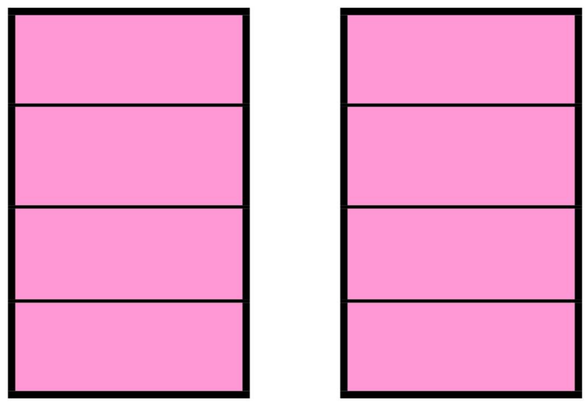


Parameter passing:

```
struct student {
   string name;
   PNG face;
   int age;
};
```

Function defn

```
int birthday(student  s) {
    s.age++;
   cout << " birthday!!! " << endl;
   return  s.age;
}
```



Example of use

```
int main() {
   student  a;
   // initialize a
   int new_age = birthday(a);
   cout <<  a.age << endl;

   return 0;
}
```

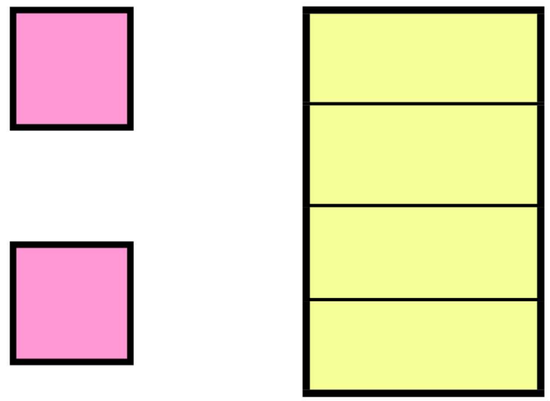
Good or bad?

Parameter passing:

```
struct student {
   string name;
   PNG face;
   int age;
};
```

Function defn

```
int birthday(student   s) {
    s.age++;
   cout << " birthday!!! " << endl;
   return   s.age;
}
```



Example of use

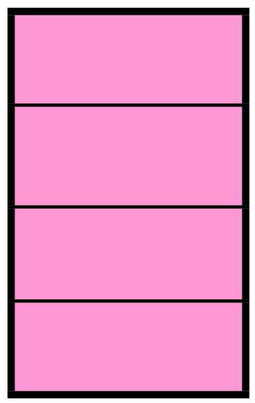
```
int main() {
   student  b;
   // initialize b
   int new_age = birthday(b);
   cout <<  b.age << endl;

   return 0;
}
```

Parameter passing:

```
struct student {
  string name;
  PNG face;
  int age;
};
```

Function defn

```
int birthday(student  s) {
    s.age++;
  cout << " birthday!!! " << endl;
  return  s.age;
}
```



Example of use

```
int main() {
  student  b;
  // initialize b
  int new_age = birthday(b);
  cout <<  b.age << endl;

  return 0;
}
```

Return values:

```
struct student {
  string name;
  PNG face;
  int age;
};
```

What happens when we run code like this:

```
int main() {
  student b;
  int new_age = birthday(b);
  return 0;
}
```

?

```
int birthday(student s) {
  s.age++;
  cout << "A birthday!!! " << s.age << endl;
  return s.age;
}
```

Return by _____ or _____ or _____ .

Return by_____

```
#include <iostream>
using namespace std;

struct student {
   string name;
   int age;
};

student * birthday(student s) {
   student y;
   student w = s;
   w.age++;
   cout << w.name << "'s birthday!!! " << endl;
   return &w;
}

int main() {
   student b;
   b.name = "Bill";
   b.age = 20;
   student * d;

   d = birthday(b);
23 cout << "Bill was " << b.age << " years old" << endl;
24 cout << "Bill is now " << d->age << " years old" << endl;

   return 0;
}
```

What is the output of the main function in:

line 23?

(a) 20 (b) 21 (c) garbage

line 24?

(a) 20 (b) 21 (c) garbage

Return by _____

What is the output of the main function in:

```
#include <iostream>
using namespace std;

struct student {
   string name;
   int age;
};

student & birthday(student s) {
   student y;
   student w = s;
   w.age++;
   cout << w.age << "'s birthday!!! " << endl;
   return w;
}

int main() {
   student b,d;
   b.name = "Bill";
   b.age = 20;

   d = birthday(b);
23 cout << "Bill was " << b.age << " years old" << endl;
24 cout << "Bill is now " << d.age << " years old" << endl;

   return 0;
}
```

line 23?

(a) 20 (b) 21 (c) garbage

line 24?

(a) 20 (b) 21 (c) garbage

Constructors reprise

```
class student {
public:
  student() {
    cout << "default constructor" << endl;
  }
  student(string n, int a){
    name = n;
    age = a;
    cout << "parameter constructor" << endl;
  }
  ~student(){
    cout << "destructor" << endl;
  }
  string name;
  int age;
};
```

```
student &birthday(student &s) {
  s.age++;
  return s;
}
```

```
int main() {
  student b("Bill", 40);
  student a;

  a = birthday(b);

  return 0;
}
```

How many times default constructor is called?

How many times parameter constructor is called?

How many times destructor is called?