Properties

Meet Sherlock Holmes, a world-renowned detective from London:

```
let sherlock = {
  surname: 'Holmes',
  address: { city: 'London' }
};
```

Meet Sherlock Holmes, a world-renowned detective from London:

```
let sherlock = {
  surname: 'Holmes',
  address: { city: 'London' }
};
```

His friend John Watson has recently moved in to live with Sherlock:

```
let john = {
  surname: 'Watson',
  address: sherlock.address
};
```

```
let sherlock = {
  surname: 'Holmes',
  address: { city: 'London' }
};

let john = {
  surname: 'Watson',
  address: sherlock.address
};

john.surname = 'Lennon';

john.address.city = 'Malibu';
```

Write down your answers to these questions:

```
console.log(sherlock.surname); // ?
console.log(sherlock.address.city); //
?
console.log(john.surname); // ?
console.log(john.address.city); // ?
```

```
let sherlock = {
  surname: 'Holmes',
  address: { city: 'London' }
};

let john = {
  surname: 'Watson',
  address: sherlock.address
};

john.surname = 'Lennon';

john.address.city = 'Malibu';
```

```
// "Holmes"
console.log(sherlock.surname);

// "Malibu"
console.log(sherlock.address.city);

// "Lennon"
console.log(john.surname);

// "Malibu"
console.log(john.address.city);
```

Thing to Remember

When reading `obj.prop`, if obj doesn't have a prop property, JavaScript will look for `obj.__proto__.prop`.

When writing to `obj.prop`, JavaScript will usually write to the object directly instead of traversing the prototype chain.

We can use `obj.hasOwnProperty('prop')` to determine whether our object has an own property called prop.

We can "*pollute*" a prototype by mutating it.

Exercises

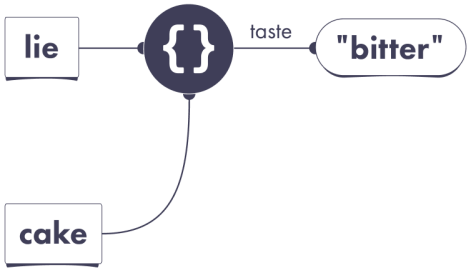
Sketch a diagram of variables and values **after** this snippet of code runs.

```
let lie = {
  taste: 'bitter'
};

let cake = {
  __proto__: lie
};
```

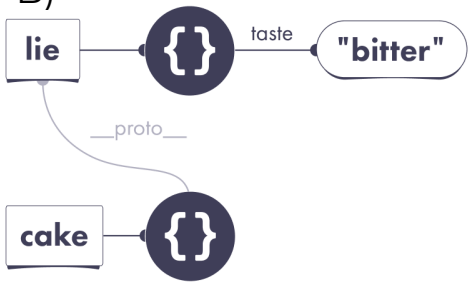
Which diagram matches your drawing the best?

```
let lie = {
  taste: 'bitter'
};

let cake = {
  __proto__: lie
};
```

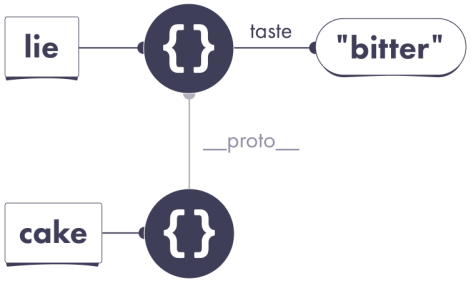
A)



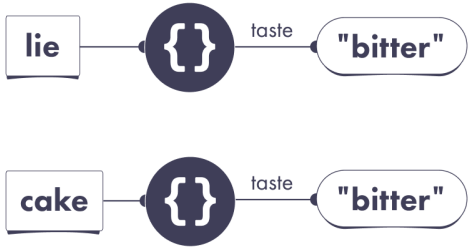
B)



C)



D)

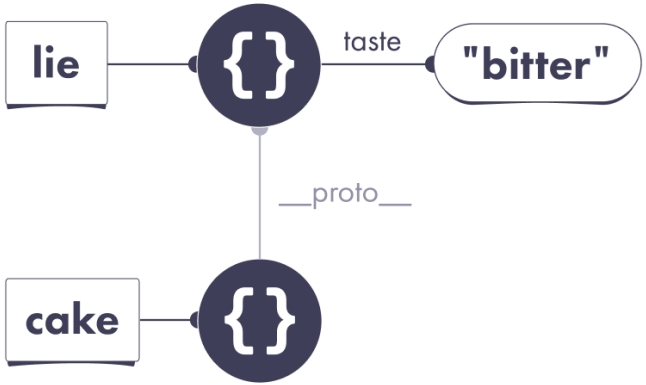


Use this diagram to answer these three questions:

1. console.log(cake === lie)

2. console.log(cake.taste === lie.taste)

3. cake.hasOwnProperty('taste') === lie.hasOwnProperty('taste')



Exercises

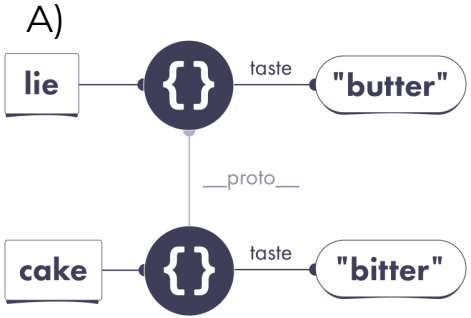
Sketch a diagram of variables and values after this snippet of code runs. Use our mental model.

```
let lie = {
  taste: 'bitter'
};

let cake = {
  __proto__: lie
};

lie.taste = 'butter';
```

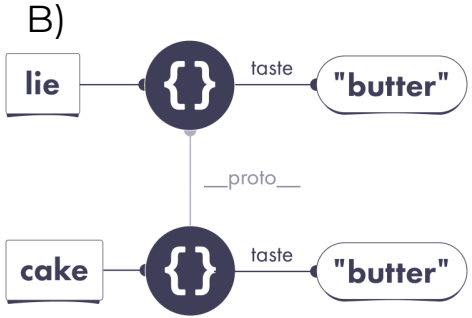
Which diagram matches your drawing the most?

```
let lie = {
  taste: 'bitter'
};

let cake = {
  __proto__: lie
};

lie.taste = 'butter';
```

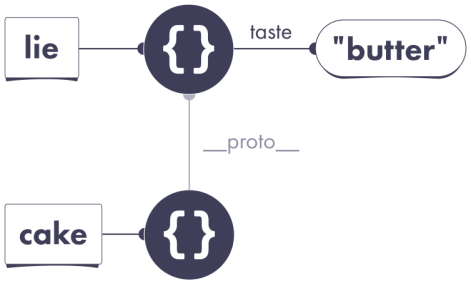
A)



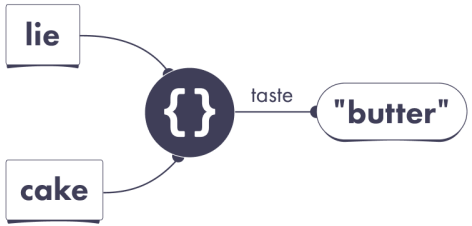
B)



C)



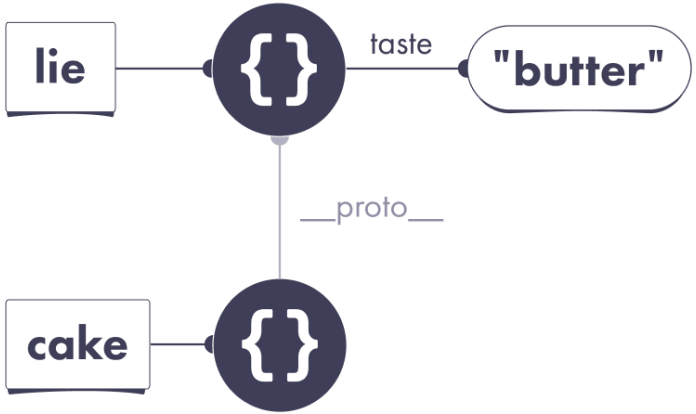
D)



Use this diagram to answer these two questions:

1. console.log(lie.taste)

2. console.log(cake.taste)



Exercises

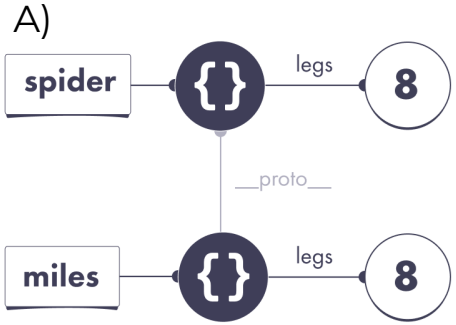
Sketch a diagram of variables and values after this snippet of code runs. Use our mental model.

```
let spider = {
  legs: 8
};
let miles = {
  __proto__: spider
};

miles.legs = 2;
```

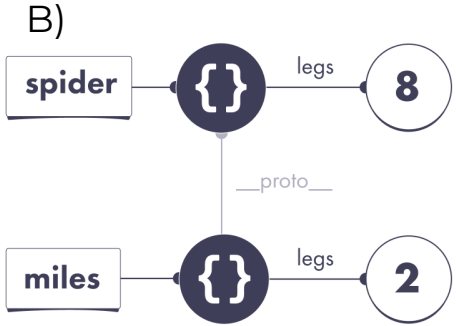
Which diagram matches your drawing the most?

```
let spider = {
  legs: 8
};
let miles = {
  __proto__: spider
};

miles.legs = 2;
```

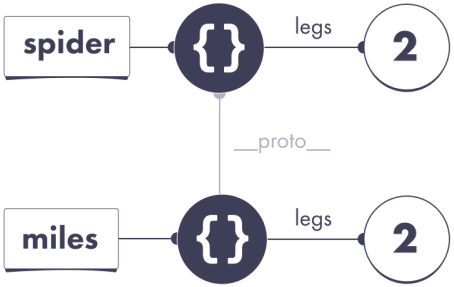
A)



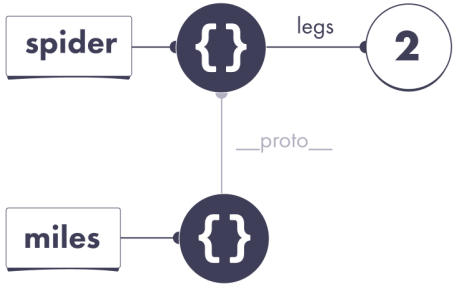
B)



C)



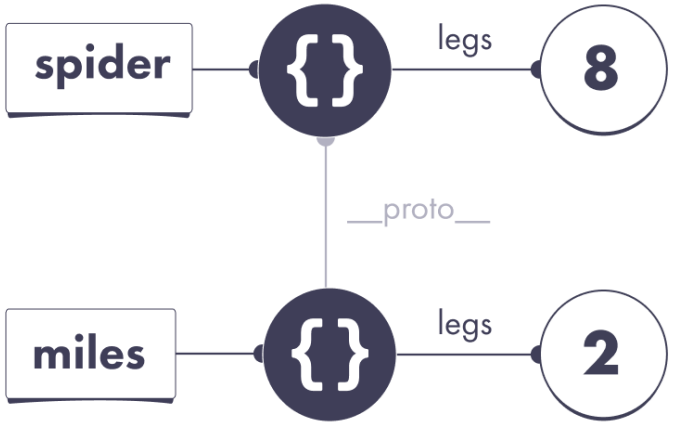
D)



Use this diagram to answer these two questions:

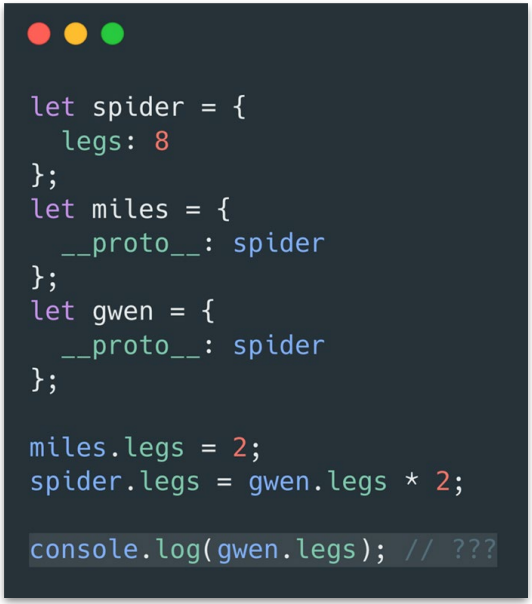
1. console.log(spider.legs)

2. console.log(miles.legs)



Exercises

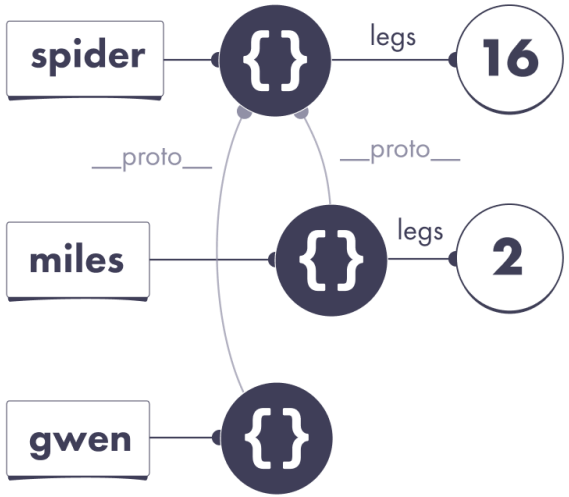
Sketch a diagram of variables and values after this snippet of code runs. Use our mental model.



```
let spider = {
  legs: 8
};
let miles = {
  __proto__: spider
};
let gwen = {
  __proto__: spider
};

miles.legs = 2;
spider.legs = gwen.legs * 2;

console.log(gwen.legs); // ???
```

Answer: 16 is the correct answer.

The object that gwen points at doesn't have a legs property, so we continue the search on its prototype. We find the legs property there, pointing at 16. That's our answer.



The second line of this code is a mystery. You have two tasks:

1. Draw the universe right after the second line
2. Figure out how the second line really ends

After you're finished, **write the ??? part in the answer field.**