Google

@fhinkel

8.1 OrdinaryGet (*O*, *P*, *Receiver*)

When the abstract operation OrdinaryGet is called with Object *O*, property key *P*, a ECMAScript language value *Receiver*, the following steps are taken:

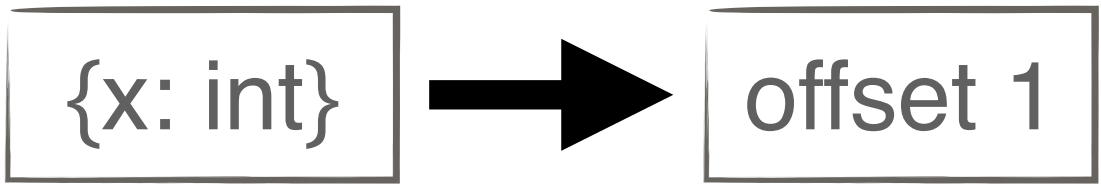
1. Assert: IsPropertyKey(*P*) is **true**.
2. Let *desc* be ? *O*.[[GetOwnProperty]](*P*).
3. If *desc* is **undefined**, then
    a. Let *parent* be ? *O*.[[GetPrototypeOf]]().
    b. If *parent* is **null**, return **undefined**.
    c. Return ? *parent*.[[Get]](*P*, *Receiver*).
4. If IsDataDescriptor(*desc*) is **true**, return *desc*.[[Value]].
5. Assert: IsAccessorDescriptor(*desc*) is **true**.
6. Let *getter* be *desc*.[[Get]].
7. If *getter* is **undefined**, return **undefined**.
8. Return ? Call(*getter*, *Receiver*).

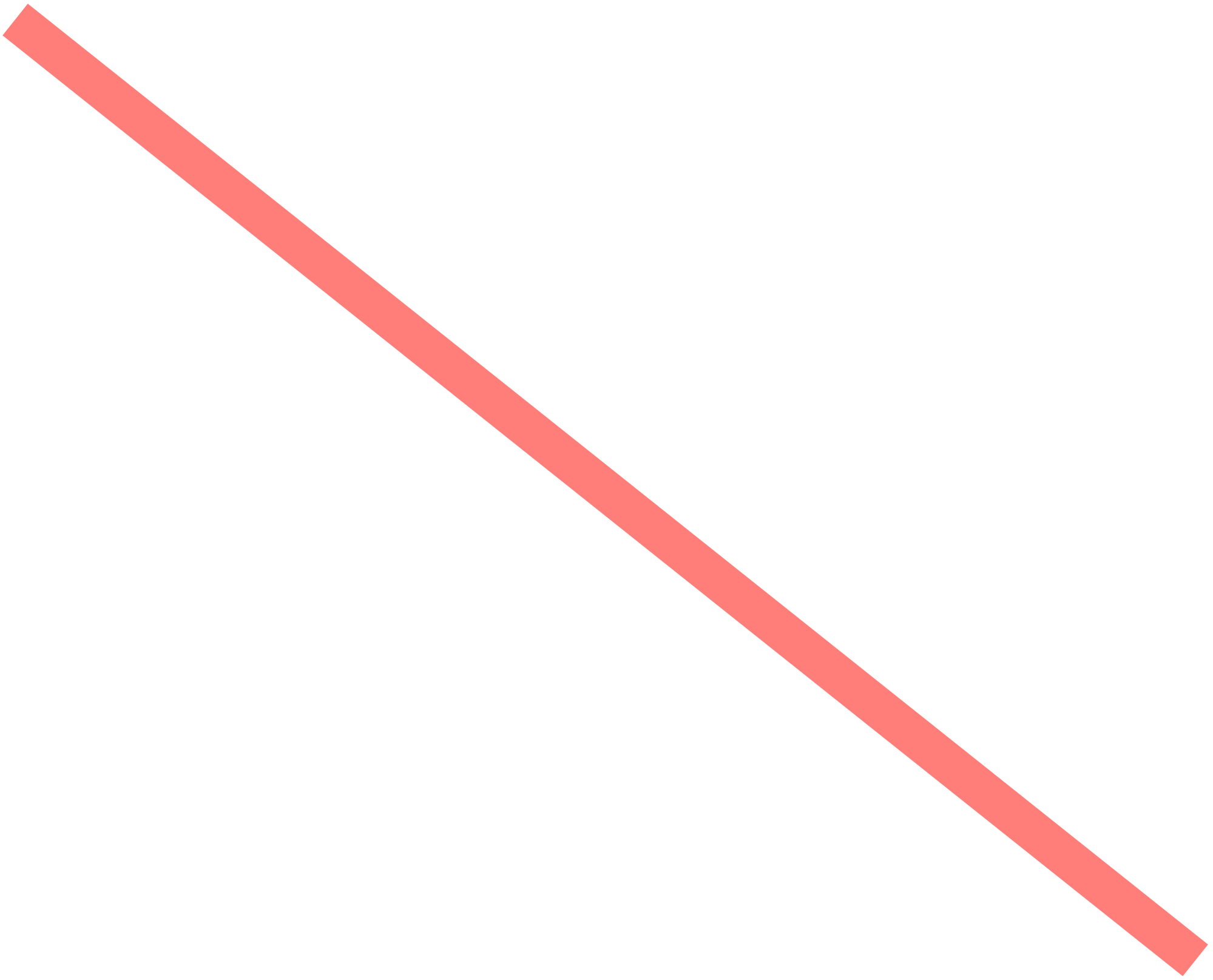
```
function add(obj) {
  return 1 + obj.x;
}
```

```
add({x: 42});
```

```
add({x: 7});
```

Cache of shortcuts





AST

Baseline Compiler

Optimizing Compiler

Machine code

Optimized Machine Code

```
function add(obj) {
  return 1 + obj.x;
}
```

Cache of shortcuts

{x: int} → offset 1

```
add({x: 42});
```

```
add({x: 7});
```

8.1 OrdinaryGet (*O*, *P*, *Receiver*) #

When the abstract operation OrdinaryGet is called with Object *O*, property key *P*, and ECMAScript language value *Receiver*, the following steps are taken:

1. Assert: IsPropertyKey(*P*) is **true**.
2. Let *desc* be ? *O*.[[GetOwnProperty]](*P*).
3. If *desc* is **undefined**, then
   a. Let *parent* be ? *O*.[[GetPrototypeOf]]().
   b. If *parent* is **null**, return **undefined**.
   c. Return ? *parent*.[[Get]](*P*, *Receiver*).
4. If IsDataDescriptor(*desc*) is **true**, return *desc*.[[Value]].
5. Assert: IsAccessorDescriptor(*desc*) is **true**.
6. Let *getter* be *desc*.[[Get]].
7. If *getter* is **undefined**, return **undefined**.
8. Return ? Call(*getter*, *Receiver*).