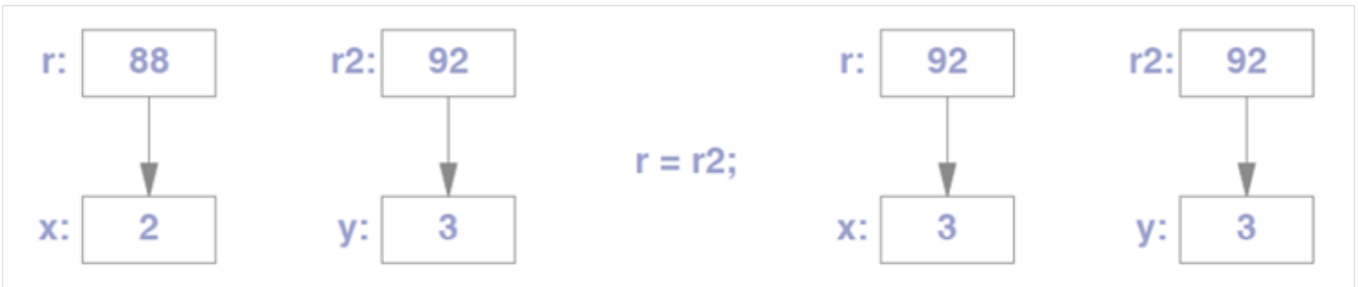

[p.22 : 본문 6행]

〉〉('~에 쓰기')는

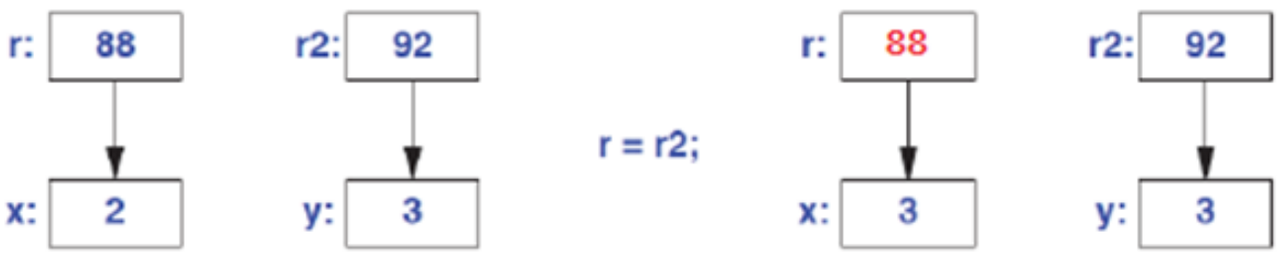
–〉

〈〈('~에 쓰기')는

[p.45]



–〉



[p.169]

for(*n)

–〉

for(--n)

[p.170]

```
template〈typename T〉
concept Equality_comparable =
    requires (T a, T b) {
  { a == b } ?〉 bool; // ==로 T를 비교
  { a != b } ?〉 bool; // !=로 T를 비교
};
```

–〉

```
template<typename T>
concept Equality_comparable =
    requires (T a, T b) {
  { a == b } -> bool; // ==로 T를 비교
  { a != b } -> bool; // !=로 T를 비교
};
```

[p.171]

```
template<typename T, typename T2 =T>
concept Equality_comparable =
    requires (T a, T2 b) {
  { a == b } ?> bool; //==로T와T2를 비교
  { a != b } ?> bool; //!=로T와T2를 비교
  { b == a } ?> bool; // ==로 T2와 T를 비교
  { b != a } ?> bool; // !=로 T2와 T를 비교
};

->

template<typename T, typename T2 =T>
concept Equality_comparable =
    requires (T a, T2 b) {
  { a == b } -> bool; //==로T와T2를 비교
  { a != b } -> bool; //!=로T와T2를 비교
  { b == a } -> bool; // ==로 T2와 T를 비교
  { b != a } -> bool; // !=로 T2와 T를 비교
};
```

[p.172]

```
  { begin(a) } ?> Iterator_type<S>; // begin(a)는 반복자를 반환
  { end(a) } ?> Iterator_type<S>; // end(a)는 반복자를 반환

  requires Same_type<Value_type<S>,Value_type<Iterator_type<S>>>;
  requires Input_iterator<Iterator_type<S>>;
};

->

{ begin(a) } -> Iterator_type<S>; // begin(a)는 반복자를 반환
  { end(a) } -> Iterator_type<S>; // end(a)는 반복자를 반환

  requires Same_type<Value_type<S>,Value_type<Iterator_type<S>>>;
  requires Input_iterator<Iterator_type<S>>;
};
```

[p.179]

```
template<typename Res, typename... Ts>
vector<Res> to_vector(Ts&&... ts)
{
vector<Res> res;
(res.push_back(ts) ...); //
return res;
}
```

–>

```
template<typename Res, typename... Ts>
vector<Res> to_vector(Ts&&... ts)
{
vector<Res> res;
(res.push_back(ts), ...); //
return res;
}
```

[p.185]

지워하는

–>

지원하는

[p.248 : 6행]

nordered_map

–>

unordered_map

[p.301 : 본문 2행]

단편화를 유발해했다.

–〉

단편화를 유발했다

[p.330 : 9행]

호출자와 동식에 동작하는

–〉

호출자와 동시에 동작하는