Trait core::cmp::PartialEq

Trait for equality comparisons which are partial equivalence relations.

This trait allows for partial equality, for types that do not have a full equivalence relation. For example, in floating point numbers `NaN != NaN`, so floating point types implement `PartialEq` but not `Eq`.

Formally, the equality must be (for all `a`, `b` and `c`):

- symmetric: `a == b` implies `b == a`; and
- transitive: `a == b` and `b == c` implies `a == c`.

Note that these requirements mean that the trait itself must be implemented symmetrically and transitively: if `T: PartialEq<U>` and `U: PartialEq<V>` then `U: PartialEq<T>` and `T: PartialEq<V>`.

Derivable

This trait can be used with `#[derive]`. When `derive`d on structs, two instances are equal if all fields are equal, and not equal if any fields are not equal. When `derive`d on enums, each variant is equal to itself and not equal to the other variants.

How can I implement `PartialEq`?

PartialEq only requires the `eq` method to be implemented; `ne` is defined in terms of it by default. Any manual implementation of `ne` *must* respect the rule that `eq` is a strict inverse of `ne`; that is, `!(a == b)` if and only if `a != b`.

Implementations of `PartialEq`, `PartialOrd`, and `Ord` *must* agree with each other. It's easy to accidentally make them disagree by deriving some of the traits and manually implementing others.

An example implementation for a domain in which two books are considered the same book if their ISBN matches, even if the formats differ:

Run
```rust
enum BookFormat { Paperback, Hardback, Ebook }
struct Book {
    isbn: i32,
    format: BookFormat,
}

impl PartialEq for Book {
    fn eq(&self, other: &Book) -> bool {
        self.isbn == other.isbn
    }
}

let b1 = Book { isbn: 3, format: BookFormat::Paperback };
let b2 = Book { isbn: 3, format: BookFormat::Ebook };
let b3 = Book { isbn: 10, format: BookFormat::Paperback };

assert!(b1 == b2);
assert!(b1 != b3);
```

Examples

Run
```rust
let x: u32 = 0;
let y: u32 = 1;

assert_eq!(x == y, false);
assert_eq!(x.eq(&y), false);
```

Required Methods

`fn eq(&self, other: &Rhs) -> bool`

This method tests for `self` and `other` values to be equal, and is used by `==`.

Provided Methods

```
fn ne(&self, other: &Rhs) -> bool
```

This method tests for `!=`.

Implementors

```
impl PartialEq for ParseFloatError                                   [src]
impl PartialEq for NonZeroU8                                         [src]
impl PartialEq for NonZeroU16                                        [src]
impl PartialEq for NonZeroU32                                        [src]
impl PartialEq for NonZeroU64                                        [src]
impl PartialEq for NonZeroU128                                       [src]
impl PartialEq for NonZeroUsize                                      [src]
impl PartialEq for NonZeroI8                                         [src]
impl PartialEq for NonZeroI16                                        [src]
impl PartialEq for NonZeroI32                                        [src]
impl PartialEq for NonZeroI64                                        [src]
impl PartialEq for NonZeroI128                                       [src]
impl PartialEq for NonZeroIsize                                      [src]
impl<T: PartialEq> PartialEq for Wrapping<T>                         [src]
impl PartialEq for FpCategory                                        [src]
impl PartialEq for ParseIntError                                     [src]
impl<T> PartialEq for Discriminant<T>                                [src]
impl<T: PartialEq> PartialEq for ManuallyDrop<T>                     [src]
impl<T: PartialEq + Zeroable> PartialEq for NonZero<T>               [src]
impl<T: ?Sized> PartialEq for *const T                               [src]
impl<T: ?Sized> PartialEq for *mut T                                 [src]
impl<Ret> PartialEq for fn() -> Ret                                  [src]
impl<Ret> PartialEq for extern "C" fn() -> Ret                       [src]
impl<Ret> PartialEq for unsafe fn() -> Ret                           [src]
impl<Ret> PartialEq for unsafe extern "C" fn() -> Ret                [src]
impl<Ret, A> PartialEq for fn(_: A) -> Ret                           [src]
impl<Ret, A> PartialEq for extern "C" fn(_: A) -> Ret                [src]
impl<Ret, A> PartialEq for extern "C" fn(_: A, ...) -> Ret           [src]
impl<Ret, A> PartialEq for unsafe fn(_: A) -> Ret                    [src]
impl<Ret, A> PartialEq for unsafe extern "C" fn(_: A) -> Ret         [src]
impl<Ret, A> PartialEq for unsafe extern "C" fn(_: A, ...) -> Ret    [src]
impl<Ret, A, B> PartialEq for fn(_: A, _: B) -> Ret                  [src]
impl<Ret, A, B> PartialEq for extern "C" fn(_: A, _: B) -> Ret       [src]
```

```
impl<Ret, A, B> PartialEq for extern "C" fn(_: A, _: B, ...) -> Ret                    [src]
impl<Ret, A, B> PartialEq for unsafe fn(_: A, _: B) -> Ret                             [src]
impl<Ret, A, B> PartialEq for unsafe extern "C" fn(_: A, _: B) -> Ret                  [src]
impl<Ret, A, B> PartialEq for unsafe extern "C" fn(_: A, _: B, ...) -> Ret             [src]
impl<Ret, A, B, C> PartialEq for fn(_: A, _: B, _: C) -> Ret                           [src]
impl<Ret, A, B, C> PartialEq for extern "C" fn(_: A, _: B, _: C) -> Ret                [src]
impl<Ret, A, B, C> PartialEq for extern "C" fn(_: A, _: B, _: C, ...) -> Ret           [src]
impl<Ret, A, B, C> PartialEq for unsafe fn(_: A, _: B, _: C) -> Ret                    [src]
impl<Ret, A, B, C> PartialEq for unsafe extern "C" fn(_: A, _: B, _: C) -> Ret         [src]
impl<Ret, A, B, C> PartialEq for unsafe extern "C" fn(_: A, _: B, _: C, ...) ->        [src]
Ret
impl<Ret, A, B, C, D> PartialEq for fn(_: A, _: B, _: C, _: D) -> Ret                  [src]
impl<Ret, A, B, C, D> PartialEq for extern "C" fn(_: A, _: B, _: C, _: D) -> Ret       [src]
impl<Ret, A, B, C, D> PartialEq for extern "C" fn(_: A, _: B, _: C, _: D, ...) ->      [src]
Ret
impl<Ret, A, B, C, D> PartialEq for unsafe fn(_: A, _: B, _: C, _: D) -> Ret           [src]
impl<Ret, A, B, C, D> PartialEq for unsafe extern "C" fn(_: A, _: B, _: C, _: D)       [src]
-> Ret
impl<Ret, A, B, C, D> PartialEq for unsafe extern "C" fn(_: A, _: B, _: C, _: D,       [src]
...) -> Ret
impl<Ret, A, B, C, D, E> PartialEq for fn(_: A, _: B, _: C, _: D, _: E) -> Ret         [src]
impl<Ret, A, B, C, D, E> PartialEq for extern "C" fn(_: A, _: B, _: C, _: D, _:        [src]
E) -> Ret
impl<Ret, A, B, C, D, E> PartialEq for extern "C" fn(_: A, _: B, _: C, _: D, _:        [src]
E, ...) -> Ret
impl<Ret, A, B, C, D, E> PartialEq for unsafe fn(_: A, _: B, _: C, _: D, _: E) ->      [src]
Ret
impl<Ret, A, B, C, D, E> PartialEq for unsafe extern "C" fn(_: A, _: B, _: C, _:       [src]
D, _: E) -> Ret
impl<Ret, A, B, C, D, E> PartialEq for unsafe extern "C" fn(_: A, _: B, _: C, _:       [src]
D, _: E, ...) -> Ret
impl<Ret, A, B, C, D, E, F> PartialEq for fn(_: A, _: B, _: C, _: D, _: E, _: F)       [src]
-> Ret
impl<Ret, A, B, C, D, E, F> PartialEq for extern "C" fn(_: A, _: B, _: C, _: D,        [src]
_: E, _: F) -> Ret
impl<Ret, A, B, C, D, E, F> PartialEq for extern "C" fn(_: A, _: B, _: C, _: D,        [src]
_: E, _: F, ...) -> Ret
impl<Ret, A, B, C, D, E, F> PartialEq for unsafe fn(_: A, _: B, _: C, _: D, _: E,      [src]
_: F) -> Ret
impl<Ret, A, B, C, D, E, F> PartialEq for unsafe extern "C" fn(_: A, _: B, _: C,       [src]
_: D, _: E, _: F) -> Ret
impl<Ret, A, B, C, D, E, F> PartialEq for unsafe extern "C" fn(_: A, _: B, _: C,       [src]
_: D, _: E, _: F, ...) -> Ret
impl<Ret, A, B, C, D, E, F, G> PartialEq for fn(_: A, _: B, _: C, _: D, _: E, _:       [src]
F, _: G) -> Ret
```

Trait PartialEq

Required Methods

eq

Provided Methods

ne

Implementors

core::cmp

Structs

Reverse

Enums

Ordering

Traits

Eq
Ord
PartialEq
PartialOrd

Functions

max
min

```
impl<Ret, A, B, C, D, E, F, G> PartialEq for extern "C" fn(_: A, _: B, _: C, _:
D, _: E, _: F, _: G) -> Ret

impl<Ret, A, B, C, D, E, F, G> PartialEq for extern "C" fn(_: A, _: B, _: C, _:   [src]
D, _: E, _: F, _: G, ...) -> Ret

impl<Ret, A, B, C, D, E, F, G> PartialEq for unsafe fn(_: A, _: B, _: C, _: D, _: [src]
E, _: F, _: G) -> Ret

impl<Ret, A, B, C, D, E, F, G> PartialEq for unsafe extern "C" fn(_: A, _: B, _:  [src]
C, _: D, _: E, _: F, _: G) -> Ret

impl<Ret, A, B, C, D, E, F, G> PartialEq for unsafe extern "C" fn(_: A, _: B, _:  [src]
C, _: D, _: E, _: F, _: G, ...) -> Ret

impl<Ret, A, B, C, D, E, F, G, H> PartialEq for fn(_: A, _: B, _: C, _: D, _: E,  [src]
_: F, _: G, _: H) -> Ret

impl<Ret, A, B, C, D, E, F, G, H> PartialEq for extern "C" fn(_: A, _: B, _: C,   [src]
_: D, _: E, _: F, _: G, _: H) -> Ret

impl<Ret, A, B, C, D, E, F, G, H> PartialEq for extern "C" fn(_: A, _: B, _: C,   [src]
_: D, _: E, _: F, _: G, _: H, ...) -> Ret

impl<Ret, A, B, C, D, E, F, G, H> PartialEq for unsafe fn(_: A, _: B, _: C, _: D, [src]
_: E, _: F, _: G, _: H) -> Ret

impl<Ret, A, B, C, D, E, F, G, H> PartialEq for unsafe extern "C" fn(_: A, _: B,  [src]
_: C, _: D, _: E, _: F, _: G, _: H) -> Ret

impl<Ret, A, B, C, D, E, F, G, H> PartialEq for unsafe extern "C" fn(_: A, _: B,  [src]
_: C, _: D, _: E, _: F, _: G, _: H, ...) -> Ret

impl<Ret, A, B, C, D, E, F, G, H, I> PartialEq for fn(_: A, _: B, _: C, _: D, _:  [src]
E, _: F, _: G, _: H, _: I) -> Ret

impl<Ret, A, B, C, D, E, F, G, H, I> PartialEq for extern "C" fn(_: A, _: B, _:   [src]
C, _: D, _: E, _: F, _: G, _: H, _: I) -> Ret

impl<Ret, A, B, C, D, E, F, G, H, I> PartialEq for extern "C" fn(_: A, _: B, _:   [src]
C, _: D, _: E, _: F, _: G, _: H, _: I, ...) -> Ret

impl<Ret, A, B, C, D, E, F, G, H, I> PartialEq for unsafe fn(_: A, _: B, _: C, _: [src]
D, _: E, _: F, _: G, _: H, _: I) -> Ret

impl<Ret, A, B, C, D, E, F, G, H, I> PartialEq for unsafe extern "C" fn(_: A, _:  [src]
B, _: C, _: D, _: E, _: F, _: G, _: H, _: I) -> Ret

impl<Ret, A, B, C, D, E, F, G, H, I> PartialEq for unsafe extern "C" fn(_: A, _:  [src]
B, _: C, _: D, _: E, _: F, _: G, _: H, _: I, ...) -> Ret

impl<Ret, A, B, C, D, E, F, G, H, I, J> PartialEq for fn(_: A, _: B, _: C, _: D,  [src]
_: E, _: F, _: G, _: H, _: I, _: J) -> Ret

impl<Ret, A, B, C, D, E, F, G, H, I, J> PartialEq for extern "C" fn(_: A, _: B,   [src]
_: C, _: D, _: E, _: F, _: G, _: H, _: I, _: J) -> Ret

impl<Ret, A, B, C, D, E, F, G, H, I, J> PartialEq for extern "C" fn(_: A, _: B,   [src]
_: C, _: D, _: E, _: F, _: G, _: H, _: I, _: J, ...) -> Ret

impl<Ret, A, B, C, D, E, F, G, H, I, J> PartialEq for unsafe fn(_: A, _: B, _: C, [src]
_: D, _: E, _: F, _: G, _: H, _: I, _: J) -> Ret

impl<Ret, A, B, C, D, E, F, G, H, I, J> PartialEq for unsafe extern "C" fn(_: A,  [src]
_: B, _: C, _: D, _: E, _: F, _: G, _: H, _: I, _: J) -> Ret

impl<Ret, A, B, C, D, E, F, G, H, I, J> PartialEq for unsafe extern "C" fn(_: A,  [src]
_: B, _: C, _: D, _: E, _: F, _: G, _: H, _: I, _: J, ...) -> Ret

impl<Ret, A, B, C, D, E, F, G, H, I, J, K> PartialEq for fn(_: A, _: B, _: C, _:  [src]
D, _: E, _: F, _: G, _: H, _: I, _: J, _: K) -> Ret

impl<Ret, A, B, C, D, E, F, G, H, I, J, K> PartialEq for extern "C" fn(_: A, _:   [src]
B, _: C, _: D, _: E, _: F, _: G, _: H, _: I, _: J, _: K) -> Ret
```

```
impl<Ret, A, B, C, D, E, F, G, H, I, J, K> PartialEq for extern "C" fn(_: A, _: B, _: C, _: D, _: E, _: F, _: G, _: H, _: I, _: J, _: K, ...) -> Ret    [src]
impl<Ret, A, B, C, D, E, F, G, H, I, J, K> PartialEq for unsafe fn(_: A, _: B, _: C, _: D, _: E, _: F, _: G, _: H, _: I, _: J, _: K) -> Ret    [src]
impl<Ret, A, B, C, D, E, F, G, H, I, J, K> PartialEq for unsafe extern "C" fn(_: A, _: B, _: C, _: D, _: E, _: F, _: G, _: H, _: I, _: J, _: K) -> Ret    [src]
impl<Ret, A, B, C, D, E, F, G, H, I, J, K> PartialEq for unsafe extern "C" fn(_: A, _: B, _: C, _: D, _: E, _: F, _: G, _: H, _: I, _: J, _: K, ...) -> Ret    [src]
impl<Ret, A, B, C, D, E, F, G, H, I, J, K, L> PartialEq for fn(_: A, _: B, _: C, _: D, _: E, _: F, _: G, _: H, _: I, _: J, _: K, _: L) -> Ret    [src]
impl<Ret, A, B, C, D, E, F, G, H, I, J, K, L> PartialEq for extern "C" fn(_: A, _: B, _: C, _: D, _: E, _: F, _: G, _: H, _: I, _: J, _: K, _: L) -> Ret    [src]
impl<Ret, A, B, C, D, E, F, G, H, I, J, K, L> PartialEq for extern "C" fn(_: A, _: B, _: C, _: D, _: E, _: F, _: G, _: H, _: I, _: J, _: K, _: L, ...) -> Ret    [src]
impl<Ret, A, B, C, D, E, F, G, H, I, J, K, L> PartialEq for unsafe fn(_: A, _: B, _: C, _: D, _: E, _: F, _: G, _: H, _: I, _: J, _: K, _: L) -> Ret    [src]
impl<Ret, A, B, C, D, E, F, G, H, I, J, K, L> PartialEq for unsafe extern "C" fn(_: A, _: B, _: C, _: D, _: E, _: F, _: G, _: H, _: I, _: J, _: K, _: L) -> Ret    [src]
impl<Ret, A, B, C, D, E, F, G, H, I, J, K, L> PartialEq for unsafe extern "C" fn(_: A, _: B, _: C, _: D, _: E, _: F, _: G, _: H, _: I, _: J, _: K, _: L, ...) -> Ret    [src]
impl<T: ?Sized> PartialEq for NonNull<T>    [src]
impl<T: ?Sized> PartialEq for PhantomData<T>    [src]
impl<Y: PartialEq, R: PartialEq> PartialEq for GeneratorState<Y, R>    [src]
impl PartialEq for RangeFull    [src]
impl<Idx: PartialEq> PartialEq for Range<Idx>    [src]
impl<Idx: PartialEq> PartialEq for RangeFrom<Idx>    [src]
impl<Idx: PartialEq> PartialEq for RangeTo<Idx>    [src]
impl<Idx: PartialEq> PartialEq for RangeInclusive<Idx>    [src]
impl<Idx: PartialEq> PartialEq for RangeToInclusive<Idx>    [src]
impl<T: PartialEq> PartialEq for Bound<T>    [src]
impl PartialEq for ()    [src]
impl PartialEq for bool    [src]
impl PartialEq for char    [src]
impl PartialEq for usize    [src]
impl PartialEq for u8    [src]
impl PartialEq for u16    [src]
impl PartialEq for u32    [src]
impl PartialEq for u64    [src]
impl PartialEq for u128    [src]
impl PartialEq for isize    [src]
impl PartialEq for i8    [src]
impl PartialEq for i16
```

[src]

```
impl PartialEq for i32
```
[src]

```
impl PartialEq for i64
```
[src]

```
impl PartialEq for i128
```
[src]

```
impl PartialEq for f32
```
[src]

```
impl PartialEq for f64
```
[src]

```
impl PartialEq for !
```
[src]

```
impl<'a, 'b, A: ?Sized, B: ?Sized> PartialEq<&'b B> for &'a A
where
    A: PartialEq<B>,
```
[src]

```
impl<'a, 'b, A: ?Sized, B: ?Sized> PartialEq<&'b mut B> for &'a mut A
where
    A: PartialEq<B>,
```
[src]

```
impl<'a, 'b, A: ?Sized, B: ?Sized> PartialEq<&'b mut B> for &'a A
where
    A: PartialEq<B>,
```
[src]

```
impl<'a, 'b, A: ?Sized, B: ?Sized> PartialEq<&'b B> for &'a mut A
where
    A: PartialEq<B>,
```
[src]

```
impl PartialEq for Ordering
```
[src]

```
impl<T: PartialEq> PartialEq for Reverse<T>
```
[src]

```
impl PartialEq for TypeId
```
[src]

```
impl<'a, 'b, A: Sized, B> PartialEq<[B; 0]> for [A; 0]
where
    A: PartialEq<B>,
```
[src]

```
impl<'a, 'b, A: Sized, B> PartialEq<[B]> for [A; 0]
where
    A: PartialEq<B>,
```
[src]

```
impl<'a, 'b, A: Sized, B> PartialEq<[A; 0]> for [B]
where
    B: PartialEq<A>,
```
[src]

```
impl<'a, 'b, A: Sized, B> PartialEq<&'b [B]> for [A; 0]
where
    A: PartialEq<B>,
```
[src]

```
impl<'a, 'b, A: Sized, B> PartialEq<[A; 0]> for &'b [B]
where
    B: PartialEq<A>,
```
[src]

```
impl<'a, 'b, A: Sized, B> PartialEq<&'b mut [B]> for [A; 0]
where
    A: PartialEq<B>,
```
[src]

```
impl<'a, 'b, A: Sized, B> PartialEq<[A; 0]> for &'b mut [B]
where
    B: PartialEq<A>,
```
[src]

```
impl<'a, 'b, A: Sized, B> PartialEq<[B; 1]> for [A; 1]
where
    A: PartialEq<B>,
```
[src]

```
impl<'a, 'b, A: Sized, B> PartialEq<[B]> for [A; 1]
where
    A: PartialEq<B>,
```
[src]

```
impl<'a, 'b, A: Sized, B> PartialEq<[A; 1]> for [B]
where
    B: PartialEq<A>,
```
[src]

```
impl<'a, 'b, A: Sized, B> PartialEq<&'b [B]> for [A; 1]
where
    A: PartialEq<B>,
```
[src]

```
impl<'a, 'b, A: Sized, B> PartialEq<[A; 1]> for &'b [B]
where
    B: PartialEq<A>,
```

```
impl<'a, 'b, A: Sized, B> PartialEq<&'b mut [B]> for [A; 1]
where
    A: PartialEq<B>,
```
[src]

```
impl<'a, 'b, A: Sized, B> PartialEq<[A; 1]> for &'b mut [B]
where
    B: PartialEq<A>,
```
[src]

```
impl<'a, 'b, A: Sized, B> PartialEq<[B; 2]> for [A; 2]
where
    A: PartialEq<B>,
```
[src]

```
impl<'a, 'b, A: Sized, B> PartialEq<[B]> for [A; 2]
where
    A: PartialEq<B>,
```
[src]

```
impl<'a, 'b, A: Sized, B> PartialEq<[A; 2]> for [B]
where
    B: PartialEq<A>,
```
[src]

```
impl<'a, 'b, A: Sized, B> PartialEq<&'b [B]> for [A; 2]
where
    A: PartialEq<B>,
```
[src]

```
impl<'a, 'b, A: Sized, B> PartialEq<[A; 2]> for &'b [B]
where
    B: PartialEq<A>,
```
[src]

```
impl<'a, 'b, A: Sized, B> PartialEq<&'b mut [B]> for [A; 2]
where
    A: PartialEq<B>,
```
[src]

```
impl<'a, 'b, A: Sized, B> PartialEq<[A; 2]> for &'b mut [B]
where
    B: PartialEq<A>,
```
[src]

```
impl<'a, 'b, A: Sized, B> PartialEq<[B; 3]> for [A; 3]
where
    A: PartialEq<B>,
```
[src]

```
impl<'a, 'b, A: Sized, B> PartialEq<[B]> for [A; 3]
where
    A: PartialEq<B>,
```
[src]

```
impl<'a, 'b, A: Sized, B> PartialEq<[A; 3]> for [B]
where
    B: PartialEq<A>,
```
[src]

```
impl<'a, 'b, A: Sized, B> PartialEq<&'b [B]> for [A; 3]
where
    A: PartialEq<B>,
```
[src]

```
impl<'a, 'b, A: Sized, B> PartialEq<[A; 3]> for &'b [B]
where
    B: PartialEq<A>,
```
[src]

```
impl<'a, 'b, A: Sized, B> PartialEq<&'b mut [B]> for [A; 3]
where
    A: PartialEq<B>,
```
[src]

```
impl<'a, 'b, A: Sized, B> PartialEq<[A; 3]> for &'b mut [B]
where
    B: PartialEq<A>,
```
[src]

```
impl<'a, 'b, A: Sized, B> PartialEq<[B; 4]> for [A; 4]
where
    A: PartialEq<B>,
```
[src]

```
impl<'a, 'b, A: Sized, B> PartialEq<[B]> for [A; 4]
where
    A: PartialEq<B>,
```
[src]

```
impl<'a, 'b, A: Sized, B> PartialEq<[A; 4]> for [B]
where
    B: PartialEq<A>,
```
[src]

```
impl<'a, 'b, A: Sized, B> PartialEq<&'b [B]> for [A; 4]
where
```
[src]

Trait PartialEq

Required Methods

eq

Provided Methods

ne

Implementors

core::cmp

Structs

Reverse

Enums

Ordering

Traits

Eq
Ord
PartialEq
PartialOrd

Functions

max
min

```
    A: PartialEq<B>,
```

```
impl<'a, 'b, A: Sized, B> PartialEq<[A; 4]> for &'b [B]
where
    B: PartialEq<A>,
```
[src]

```
impl<'a, 'b, A: Sized, B> PartialEq<&'b mut [B]> for [A; 4]
where
    A: PartialEq<B>,
```
[src]

```
impl<'a, 'b, A: Sized, B> PartialEq<[A; 4]> for &'b mut [B]
where
    B: PartialEq<A>,
```
[src]

```
impl<'a, 'b, A: Sized, B> PartialEq<[B; 5]> for [A; 5]
where
    A: PartialEq<B>,
```
[src]

```
impl<'a, 'b, A: Sized, B> PartialEq<[B]> for [A; 5]
where
    A: PartialEq<B>,
```
[src]

```
impl<'a, 'b, A: Sized, B> PartialEq<[A; 5]> for [B]
where
    B: PartialEq<A>,
```
[src]

```
impl<'a, 'b, A: Sized, B> PartialEq<&'b [B]> for [A; 5]
where
    A: PartialEq<B>,
```
[src]

```
impl<'a, 'b, A: Sized, B> PartialEq<[A; 5]> for &'b [B]
where
    B: PartialEq<A>,
```
[src]

```
impl<'a, 'b, A: Sized, B> PartialEq<&'b mut [B]> for [A; 5]
where
    A: PartialEq<B>,
```
[src]

```
impl<'a, 'b, A: Sized, B> PartialEq<[A; 5]> for &'b mut [B]
where
    B: PartialEq<A>,
```
[src]

```
impl<'a, 'b, A: Sized, B> PartialEq<[B; 6]> for [A; 6]
where
    A: PartialEq<B>,
```
[src]

```
impl<'a, 'b, A: Sized, B> PartialEq<[B]> for [A; 6]
where
    A: PartialEq<B>,
```
[src]

```
impl<'a, 'b, A: Sized, B> PartialEq<[A; 6]> for [B]
where
    B: PartialEq<A>,
```
[src]

```
impl<'a, 'b, A: Sized, B> PartialEq<&'b [B]> for [A; 6]
where
    A: PartialEq<B>,
```
[src]

```
impl<'a, 'b, A: Sized, B> PartialEq<[A; 6]> for &'b [B]
where
    B: PartialEq<A>,
```
[src]

```
impl<'a, 'b, A: Sized, B> PartialEq<&'b mut [B]> for [A; 6]
where
    A: PartialEq<B>,
```
[src]

```
impl<'a, 'b, A: Sized, B> PartialEq<[A; 6]> for &'b mut [B]
where
    B: PartialEq<A>,
```
[src]

```
impl<'a, 'b, A: Sized, B> PartialEq<[B; 7]> for [A; 7]
where
    A: PartialEq<B>,
```
[src]

```
impl<'a, 'b, A: Sized, B> PartialEq<[B]> for [A; 7]
where
    A: PartialEq<B>,
```
[src]

```
impl<'a, 'b, A: Sized, B> PartialEq<[A; 7]> for [B]
where
    B: PartialEq<A>,
```
[src]

```
impl<'a, 'b, A: Sized, B> PartialEq<&'b [B]> for [A; 7]
where
    A: PartialEq<B>,
```

```
impl<'a, 'b, A: Sized, B> PartialEq<[A; 7]> for &'b [B]
where
    B: PartialEq<A>,
```
[src]

```
impl<'a, 'b, A: Sized, B> PartialEq<&'b mut [B]> for [A; 7]
where
    A: PartialEq<B>,
```
[src]

```
impl<'a, 'b, A: Sized, B> PartialEq<[A; 7]> for &'b mut [B]
where
    B: PartialEq<A>,
```
[src]

```
impl<'a, 'b, A: Sized, B> PartialEq<[B; 8]> for [A; 8]
where
    A: PartialEq<B>,
```
[src]

```
impl<'a, 'b, A: Sized, B> PartialEq<[B]> for [A; 8]
where
    A: PartialEq<B>,
```
[src]

```
impl<'a, 'b, A: Sized, B> PartialEq<[A; 8]> for [B]
where
    B: PartialEq<A>,
```
[src]

```
impl<'a, 'b, A: Sized, B> PartialEq<&'b [B]> for [A; 8]
where
    A: PartialEq<B>,
```
[src]

```
impl<'a, 'b, A: Sized, B> PartialEq<[A; 8]> for &'b [B]
where
    B: PartialEq<A>,
```
[src]

```
impl<'a, 'b, A: Sized, B> PartialEq<&'b mut [B]> for [A; 8]
where
    A: PartialEq<B>,
```
[src]

```
impl<'a, 'b, A: Sized, B> PartialEq<[A; 8]> for &'b mut [B]
where
    B: PartialEq<A>,
```
[src]

```
impl<'a, 'b, A: Sized, B> PartialEq<[B; 9]> for [A; 9]
where
    A: PartialEq<B>,
```
[src]

```
impl<'a, 'b, A: Sized, B> PartialEq<[B]> for [A; 9]
where
    A: PartialEq<B>,
```
[src]

```
impl<'a, 'b, A: Sized, B> PartialEq<[A; 9]> for [B]
where
    B: PartialEq<A>,
```
[src]

```
impl<'a, 'b, A: Sized, B> PartialEq<&'b [B]> for [A; 9]
where
    A: PartialEq<B>,
```
[src]

```
impl<'a, 'b, A: Sized, B> PartialEq<[A; 9]> for &'b [B]
where
    B: PartialEq<A>,
```
[src]

```
impl<'a, 'b, A: Sized, B> PartialEq<&'b mut [B]> for [A; 9]
where
    A: PartialEq<B>,
```
[src]

```
impl<'a, 'b, A: Sized, B> PartialEq<[A; 9]> for &'b mut [B]
where
    B: PartialEq<A>,
```
[src]

```
impl<'a, 'b, A: Sized, B> PartialEq<[B; 10]> for [A; 10]
where
    A: PartialEq<B>,
```
[src]

```
impl<'a, 'b, A: Sized, B> PartialEq<[B]> for [A; 10]
where
    A: PartialEq<B>,
```
[src]

```
impl<'a, 'b, A: Sized, B> PartialEq<[A; 10]> for [B]
where
```
[src]

Trait PartialEq

Required Methods

eq

Provided Methods

ne

Implementors

core::cmp

Structs

Reverse

Enums

Ordering

Traits

Eq
Ord
PartialEq
PartialOrd

Functions

max
min

```
    B: PartialEq<A>,
```

```
impl<'a, 'b, A: Sized, B> PartialEq<&'b [B]> for [A; 10]
where
    A: PartialEq<B>,
```
[src]

```
impl<'a, 'b, A: Sized, B> PartialEq<[A; 10]> for &'b [B]
where
    B: PartialEq<A>,
```
[src]

```
impl<'a, 'b, A: Sized, B> PartialEq<&'b mut [B]> for [A; 10]
where
    A: PartialEq<B>,
```
[src]

```
impl<'a, 'b, A: Sized, B> PartialEq<[A; 10]> for &'b mut [B]
where
    B: PartialEq<A>,
```
[src]

```
impl<'a, 'b, A: Sized, B> PartialEq<[B; 11]> for [A; 11]
where
    A: PartialEq<B>,
```
[src]

```
impl<'a, 'b, A: Sized, B> PartialEq<[B]> for [A; 11]
where
    A: PartialEq<B>,
```
[src]

```
impl<'a, 'b, A: Sized, B> PartialEq<[A; 11]> for [B]
where
    B: PartialEq<A>,
```
[src]

```
impl<'a, 'b, A: Sized, B> PartialEq<&'b [B]> for [A; 11]
where
    A: PartialEq<B>,
```
[src]

```
impl<'a, 'b, A: Sized, B> PartialEq<[A; 11]> for &'b [B]
where
    B: PartialEq<A>,
```
[src]

```
impl<'a, 'b, A: Sized, B> PartialEq<&'b mut [B]> for [A; 11]
where
    A: PartialEq<B>,
```
[src]

```
impl<'a, 'b, A: Sized, B> PartialEq<[A; 11]> for &'b mut [B]
where
    B: PartialEq<A>,
```
[src]

```
impl<'a, 'b, A: Sized, B> PartialEq<[B; 12]> for [A; 12]
where
    A: PartialEq<B>,
```
[src]

```
impl<'a, 'b, A: Sized, B> PartialEq<[B]> for [A; 12]
where
    A: PartialEq<B>,
```
[src]

```
impl<'a, 'b, A: Sized, B> PartialEq<[A; 12]> for [B]
where
    B: PartialEq<A>,
```
[src]

```
impl<'a, 'b, A: Sized, B> PartialEq<&'b [B]> for [A; 12]
where
    A: PartialEq<B>,
```
[src]

```
impl<'a, 'b, A: Sized, B> PartialEq<[A; 12]> for &'b [B]
where
    B: PartialEq<A>,
```
[src]

```
impl<'a, 'b, A: Sized, B> PartialEq<&'b mut [B]> for [A; 12]
where
    A: PartialEq<B>,
```
[src]

```
impl<'a, 'b, A: Sized, B> PartialEq<[A; 12]> for &'b mut [B]
where
    B: PartialEq<A>,
```
[src]

```
impl<'a, 'b, A: Sized, B> PartialEq<[B; 13]> for [A; 13]
where
    A: PartialEq<B>,
```
[src]

```
impl<'a, 'b, A: Sized, B> PartialEq<[B]> for [A; 13]
where
    A: PartialEq<B>,
```
[src]

Trait PartialEq

Required Methods

eq

Provided Methods

ne

Implementors

core::cmp

Structs

Reverse

Enums

Ordering

Traits

Eq
Ord
PartialEq
PartialOrd

Functions

max
min

```
impl<'a, 'b, A: Sized, B> PartialEq<[A; 13]> for [B]
where
    B: PartialEq<A>,

impl<'a, 'b, A: Sized, B> PartialEq<&'b [B]> for [A; 13]              [src]
where
    A: PartialEq<B>,

impl<'a, 'b, A: Sized, B> PartialEq<[A; 13]> for &'b [B]              [src]
where
    B: PartialEq<A>,

impl<'a, 'b, A: Sized, B> PartialEq<&'b mut [B]> for [A; 13]          [src]
where
    A: PartialEq<B>,

impl<'a, 'b, A: Sized, B> PartialEq<[A; 13]> for &'b mut [B]          [src]
where
    B: PartialEq<A>,

impl<'a, 'b, A: Sized, B> PartialEq<[B; 14]> for [A; 14]              [src]
where
    A: PartialEq<B>,

impl<'a, 'b, A: Sized, B> PartialEq<[B]> for [A; 14]                  [src]
where
    A: PartialEq<B>,

impl<'a, 'b, A: Sized, B> PartialEq<[A; 14]> for [B]                  [src]
where
    B: PartialEq<A>,

impl<'a, 'b, A: Sized, B> PartialEq<&'b [B]> for [A; 14]              [src]
where
    A: PartialEq<B>,

impl<'a, 'b, A: Sized, B> PartialEq<[A; 14]> for &'b [B]              [src]
where
    B: PartialEq<A>,

impl<'a, 'b, A: Sized, B> PartialEq<&'b mut [B]> for [A; 14]          [src]
where
    A: PartialEq<B>,

impl<'a, 'b, A: Sized, B> PartialEq<[A; 14]> for &'b mut [B]          [src]
where
    B: PartialEq<A>,

impl<'a, 'b, A: Sized, B> PartialEq<[B; 15]> for [A; 15]              [src]
where
    A: PartialEq<B>,

impl<'a, 'b, A: Sized, B> PartialEq<[B]> for [A; 15]                  [src]
where
    A: PartialEq<B>,

impl<'a, 'b, A: Sized, B> PartialEq<[A; 15]> for [B]                  [src]
where
    B: PartialEq<A>,

impl<'a, 'b, A: Sized, B> PartialEq<&'b [B]> for [A; 15]              [src]
where
    A: PartialEq<B>,

impl<'a, 'b, A: Sized, B> PartialEq<[A; 15]> for &'b [B]              [src]
where
    B: PartialEq<A>,

impl<'a, 'b, A: Sized, B> PartialEq<&'b mut [B]> for [A; 15]          [src]
where
    A: PartialEq<B>,

impl<'a, 'b, A: Sized, B> PartialEq<[A; 15]> for &'b mut [B]          [src]
where
    B: PartialEq<A>,

impl<'a, 'b, A: Sized, B> PartialEq<[B; 16]> for [A; 16]              [src]
where
    A: PartialEq<B>,

impl<'a, 'b, A: Sized, B> PartialEq<[B]> for [A; 16]                  [src]
where
```

```
    A: PartialEq<B>,
```

```
impl<'a, 'b, A: Sized, B> PartialEq<[A; 16]> for [B]                  [src]
where
    B: PartialEq<A>,

impl<'a, 'b, A: Sized, B> PartialEq<&'b [B]> for [A; 16]              [src]
where
    A: PartialEq<B>,

impl<'a, 'b, A: Sized, B> PartialEq<[A; 16]> for &'b [B]              [src]
where
    B: PartialEq<A>,

impl<'a, 'b, A: Sized, B> PartialEq<&'b mut [B]> for [A; 16]          [src]
where
    A: PartialEq<B>,

impl<'a, 'b, A: Sized, B> PartialEq<[A; 16]> for &'b mut [B]          [src]
where
    B: PartialEq<A>,

impl<'a, 'b, A: Sized, B> PartialEq<[B; 17]> for [A; 17]              [src]
where
    A: PartialEq<B>,

impl<'a, 'b, A: Sized, B> PartialEq<[B]> for [A; 17]                  [src]
where
    A: PartialEq<B>,

impl<'a, 'b, A: Sized, B> PartialEq<[A; 17]> for [B]                  [src]
where
    B: PartialEq<A>,

impl<'a, 'b, A: Sized, B> PartialEq<&'b [B]> for [A; 17]              [src]
where
    A: PartialEq<B>,

impl<'a, 'b, A: Sized, B> PartialEq<[A; 17]> for &'b [B]              [src]
where
    B: PartialEq<A>,

impl<'a, 'b, A: Sized, B> PartialEq<&'b mut [B]> for [A; 17]          [src]
where
    A: PartialEq<B>,

impl<'a, 'b, A: Sized, B> PartialEq<[A; 17]> for &'b mut [B]          [src]
where
    B: PartialEq<A>,

impl<'a, 'b, A: Sized, B> PartialEq<[B; 18]> for [A; 18]              [src]
where
    A: PartialEq<B>,

impl<'a, 'b, A: Sized, B> PartialEq<[B]> for [A; 18]                  [src]
where
    A: PartialEq<B>,

impl<'a, 'b, A: Sized, B> PartialEq<[A; 18]> for [B]                  [src]
where
    B: PartialEq<A>,

impl<'a, 'b, A: Sized, B> PartialEq<&'b [B]> for [A; 18]              [src]
where
    A: PartialEq<B>,

impl<'a, 'b, A: Sized, B> PartialEq<[A; 18]> for &'b [B]              [src]
where
    B: PartialEq<A>,

impl<'a, 'b, A: Sized, B> PartialEq<&'b mut [B]> for [A; 18]          [src]
where
    A: PartialEq<B>,

impl<'a, 'b, A: Sized, B> PartialEq<[A; 18]> for &'b mut [B]          [src]
where
    B: PartialEq<A>,

impl<'a, 'b, A: Sized, B> PartialEq<[B; 19]> for [A; 19]              [src]
where
    A: PartialEq<B>,
impl<'a, 'b, A: Sized, B> PartialEq<[B]> for [A; 19]
where
    A: PartialEq<B>,
```

```
impl<'a, 'b, A: Sized, B> PartialEq<[A; 19]> for [B]
where
    B: PartialEq<A>,
```
[src]

```
impl<'a, 'b, A: Sized, B> PartialEq<&'b [B]> for [A; 19]
where
    A: PartialEq<B>,
```
[src]

```
impl<'a, 'b, A: Sized, B> PartialEq<[A; 19]> for &'b [B]
where
    B: PartialEq<A>,
```
[src]

```
impl<'a, 'b, A: Sized, B> PartialEq<&'b mut [B]> for [A; 19]
where
    A: PartialEq<B>,
```
[src]

```
impl<'a, 'b, A: Sized, B> PartialEq<[A; 19]> for &'b mut [B]
where
    B: PartialEq<A>,
```
[src]

```
impl<'a, 'b, A: Sized, B> PartialEq<[B; 20]> for [A; 20]
where
    A: PartialEq<B>,
```
[src]

```
impl<'a, 'b, A: Sized, B> PartialEq<[B]> for [A; 20]
where
    A: PartialEq<B>,
```
[src]

```
impl<'a, 'b, A: Sized, B> PartialEq<[A; 20]> for [B]
where
    B: PartialEq<A>,
```
[src]

```
impl<'a, 'b, A: Sized, B> PartialEq<&'b [B]> for [A; 20]
where
    A: PartialEq<B>,
```
[src]

```
impl<'a, 'b, A: Sized, B> PartialEq<[A; 20]> for &'b [B]
where
    B: PartialEq<A>,
```
[src]

```
impl<'a, 'b, A: Sized, B> PartialEq<&'b mut [B]> for [A; 20]
where
    A: PartialEq<B>,
```
[src]

```
impl<'a, 'b, A: Sized, B> PartialEq<[A; 20]> for &'b mut [B]
where
    B: PartialEq<A>,
```
[src]

```
impl<'a, 'b, A: Sized, B> PartialEq<[B; 21]> for [A; 21]
where
    A: PartialEq<B>,
```
[src]

```
impl<'a, 'b, A: Sized, B> PartialEq<[B]> for [A; 21]
where
    A: PartialEq<B>,
```
[src]

```
impl<'a, 'b, A: Sized, B> PartialEq<[A; 21]> for [B]
where
    B: PartialEq<A>,
```
[src]

```
impl<'a, 'b, A: Sized, B> PartialEq<&'b [B]> for [A; 21]
where
    A: PartialEq<B>,
```
[src]

```
impl<'a, 'b, A: Sized, B> PartialEq<[A; 21]> for &'b [B]
where
    B: PartialEq<A>,
```
[src]

```
impl<'a, 'b, A: Sized, B> PartialEq<&'b mut [B]> for [A; 21]
where
    A: PartialEq<B>,
```
[src]

```
impl<'a, 'b, A: Sized, B> PartialEq<[A; 21]> for &'b mut [B]
where
    B: PartialEq<A>,
```
[src]

```
impl<'a, 'b, A: Sized, B> PartialEq<[B; 22]> for [A; 22]
where
```
[src]

```
    A: PartialEq<B>,

impl<'a, 'b, A: Sized, B> PartialEq<[B]> for [A; 22]                     [src]
where
    A: PartialEq<B>,

impl<'a, 'b, A: Sized, B> PartialEq<[A; 22]> for [B]                     [src]
where
    B: PartialEq<A>,

impl<'a, 'b, A: Sized, B> PartialEq<&'b [B]> for [A; 22]                 [src]
where
    A: PartialEq<B>,

impl<'a, 'b, A: Sized, B> PartialEq<[A; 22]> for &'b [B]                 [src]
where
    B: PartialEq<A>,

impl<'a, 'b, A: Sized, B> PartialEq<&'b mut [B]> for [A; 22]             [src]
where
    A: PartialEq<B>,

impl<'a, 'b, A: Sized, B> PartialEq<[A; 22]> for &'b mut [B]             [src]
where
    B: PartialEq<A>,

impl<'a, 'b, A: Sized, B> PartialEq<[B; 23]> for [A; 23]                 [src]
where
    A: PartialEq<B>,

impl<'a, 'b, A: Sized, B> PartialEq<[B]> for [A; 23]                     [src]
where
    A: PartialEq<B>,

impl<'a, 'b, A: Sized, B> PartialEq<[A; 23]> for [B]                     [src]
where
    B: PartialEq<A>,

impl<'a, 'b, A: Sized, B> PartialEq<&'b [B]> for [A; 23]                 [src]
where
    A: PartialEq<B>,

impl<'a, 'b, A: Sized, B> PartialEq<[A; 23]> for &'b [B]                 [src]
where
    B: PartialEq<A>,

impl<'a, 'b, A: Sized, B> PartialEq<&'b mut [B]> for [A; 23]             [src]
where
    A: PartialEq<B>,

impl<'a, 'b, A: Sized, B> PartialEq<[A; 23]> for &'b mut [B]             [src]
where
    B: PartialEq<A>,

impl<'a, 'b, A: Sized, B> PartialEq<[B; 24]> for [A; 24]                 [src]
where
    A: PartialEq<B>,

impl<'a, 'b, A: Sized, B> PartialEq<[B]> for [A; 24]                     [src]
where
    A: PartialEq<B>,

impl<'a, 'b, A: Sized, B> PartialEq<[A; 24]> for [B]                     [src]
where
    B: PartialEq<A>,

impl<'a, 'b, A: Sized, B> PartialEq<&'b [B]> for [A; 24]                 [src]
where
    A: PartialEq<B>,

impl<'a, 'b, A: Sized, B> PartialEq<[A; 24]> for &'b [B]                 [src]
where
    B: PartialEq<A>,

impl<'a, 'b, A: Sized, B> PartialEq<&'b mut [B]> for [A; 24]             [src]
where
    A: PartialEq<B>,

impl<'a, 'b, A: Sized, B> PartialEq<[A; 24]> for &'b mut [B]             [src]
where
    B: PartialEq<A>,
impl<'a, 'b, A: Sized, B> PartialEq<[B; 25]> for [A; 25]
where
    A: PartialEq<B>,
```

```
impl<'a, 'b, A: Sized, B> PartialEq<[B]> for [A; 25]
where
    A: PartialEq<B>,
```
[src]

```
impl<'a, 'b, A: Sized, B> PartialEq<[A; 25]> for [B]
where
    B: PartialEq<A>,
```
[src]

```
impl<'a, 'b, A: Sized, B> PartialEq<&'b [B]> for [A; 25]
where
    A: PartialEq<B>,
```
[src]

```
impl<'a, 'b, A: Sized, B> PartialEq<[A; 25]> for &'b [B]
where
    B: PartialEq<A>,
```
[src]

```
impl<'a, 'b, A: Sized, B> PartialEq<&'b mut [B]> for [A; 25]
where
    A: PartialEq<B>,
```
[src]

```
impl<'a, 'b, A: Sized, B> PartialEq<[A; 25]> for &'b mut [B]
where
    B: PartialEq<A>,
```
[src]

```
impl<'a, 'b, A: Sized, B> PartialEq<[B; 26]> for [A; 26]
where
    A: PartialEq<B>,
```
[src]

```
impl<'a, 'b, A: Sized, B> PartialEq<[B]> for [A; 26]
where
    A: PartialEq<B>,
```
[src]

```
impl<'a, 'b, A: Sized, B> PartialEq<[A; 26]> for [B]
where
    B: PartialEq<A>,
```
[src]

```
impl<'a, 'b, A: Sized, B> PartialEq<&'b [B]> for [A; 26]
where
    A: PartialEq<B>,
```
[src]

```
impl<'a, 'b, A: Sized, B> PartialEq<[A; 26]> for &'b [B]
where
    B: PartialEq<A>,
```
[src]

```
impl<'a, 'b, A: Sized, B> PartialEq<&'b mut [B]> for [A; 26]
where
    A: PartialEq<B>,
```
[src]

```
impl<'a, 'b, A: Sized, B> PartialEq<[A; 26]> for &'b mut [B]
where
    B: PartialEq<A>,
```
[src]

```
impl<'a, 'b, A: Sized, B> PartialEq<[B; 27]> for [A; 27]
where
    A: PartialEq<B>,
```
[src]

```
impl<'a, 'b, A: Sized, B> PartialEq<[B]> for [A; 27]
where
    A: PartialEq<B>,
```
[src]

```
impl<'a, 'b, A: Sized, B> PartialEq<[A; 27]> for [B]
where
    B: PartialEq<A>,
```
[src]

```
impl<'a, 'b, A: Sized, B> PartialEq<&'b [B]> for [A; 27]
where
    A: PartialEq<B>,
```
[src]

```
impl<'a, 'b, A: Sized, B> PartialEq<[A; 27]> for &'b [B]
where
    B: PartialEq<A>,
```
[src]

```
impl<'a, 'b, A: Sized, B> PartialEq<&'b mut [B]> for [A; 27]
where
    A: PartialEq<B>,
```
[src]

```
impl<'a, 'b, A: Sized, B> PartialEq<[A; 27]> for &'b mut [B]
where
```
[src]

```
    B: PartialEq<A>,

impl<'a, 'b, A: Sized, B> PartialEq<[B; 28]> for [A; 28]                    [src]
where
    A: PartialEq<B>,

impl<'a, 'b, A: Sized, B> PartialEq<[B]> for [A; 28]                        [src]
where
    A: PartialEq<B>,

impl<'a, 'b, A: Sized, B> PartialEq<[A; 28]> for [B]                        [src]
where
    B: PartialEq<A>,

impl<'a, 'b, A: Sized, B> PartialEq<&'b [B]> for [A; 28]                    [src]
where
    A: PartialEq<B>,

impl<'a, 'b, A: Sized, B> PartialEq<[A; 28]> for &'b [B]                    [src]
where
    B: PartialEq<A>,

impl<'a, 'b, A: Sized, B> PartialEq<&'b mut [B]> for [A; 28]                [src]
where
    A: PartialEq<B>,

impl<'a, 'b, A: Sized, B> PartialEq<[A; 28]> for &'b mut [B]                [src]
where
    B: PartialEq<A>,

impl<'a, 'b, A: Sized, B> PartialEq<[B; 29]> for [A; 29]                    [src]
where
    A: PartialEq<B>,

impl<'a, 'b, A: Sized, B> PartialEq<[B]> for [A; 29]                        [src]
where
    A: PartialEq<B>,

impl<'a, 'b, A: Sized, B> PartialEq<[A; 29]> for [B]                        [src]
where
    B: PartialEq<A>,

impl<'a, 'b, A: Sized, B> PartialEq<&'b [B]> for [A; 29]                    [src]
where
    A: PartialEq<B>,

impl<'a, 'b, A: Sized, B> PartialEq<[A; 29]> for &'b [B]                    [src]
where
    B: PartialEq<A>,

impl<'a, 'b, A: Sized, B> PartialEq<&'b mut [B]> for [A; 29]                [src]
where
    A: PartialEq<B>,

impl<'a, 'b, A: Sized, B> PartialEq<[A; 29]> for &'b mut [B]                [src]
where
    B: PartialEq<A>,

impl<'a, 'b, A: Sized, B> PartialEq<[B; 30]> for [A; 30]                    [src]
where
    A: PartialEq<B>,

impl<'a, 'b, A: Sized, B> PartialEq<[B]> for [A; 30]                        [src]
where
    A: PartialEq<B>,

impl<'a, 'b, A: Sized, B> PartialEq<[A; 30]> for [B]                        [src]
where
    B: PartialEq<A>,

impl<'a, 'b, A: Sized, B> PartialEq<&'b [B]> for [A; 30]                    [src]
where
    A: PartialEq<B>,

impl<'a, 'b, A: Sized, B> PartialEq<[A; 30]> for &'b [B]                    [src]
where
    B: PartialEq<A>,

impl<'a, 'b, A: Sized, B> PartialEq<&'b mut [B]> for [A; 30]                [src]
where
    A: PartialEq<B>,
```

Trait PartialEq

Required Methods

eq

Provided Methods

ne

Implementors

core::cmp

Structs

Reverse

Enums

Ordering

Traits

Eq
Ord
PartialEq
PartialOrd

Functions

max
min

```
impl<'a, 'b, A: Sized, B> PartialEq<[A; 30]> for &'b mut [B]
where
    B: PartialEq<A>,

impl<'a, 'b, A: Sized, B> PartialEq<[B; 31]> for [A; 31]                [src]
where
    A: PartialEq<B>,

impl<'a, 'b, A: Sized, B> PartialEq<[B]> for [A; 31]                    [src]
where
    A: PartialEq<B>,

impl<'a, 'b, A: Sized, B> PartialEq<[A; 31]> for [B]                    [src]
where
    B: PartialEq<A>,

impl<'a, 'b, A: Sized, B> PartialEq<&'b [B]> for [A; 31]                [src]
where
    A: PartialEq<B>,

impl<'a, 'b, A: Sized, B> PartialEq<[A; 31]> for &'b [B]                [src]
where
    B: PartialEq<A>,

impl<'a, 'b, A: Sized, B> PartialEq<&'b mut [B]> for [A; 31]            [src]
where
    A: PartialEq<B>,

impl<'a, 'b, A: Sized, B> PartialEq<[A; 31]> for &'b mut [B]            [src]
where
    B: PartialEq<A>,

impl<'a, 'b, A: Sized, B> PartialEq<[B; 32]> for [A; 32]                [src]
where
    A: PartialEq<B>,

impl<'a, 'b, A: Sized, B> PartialEq<[B]> for [A; 32]                    [src]
where
    A: PartialEq<B>,

impl<'a, 'b, A: Sized, B> PartialEq<[A; 32]> for [B]                    [src]
where
    B: PartialEq<A>,

impl<'a, 'b, A: Sized, B> PartialEq<&'b [B]> for [A; 32]                [src]
where
    A: PartialEq<B>,

impl<'a, 'b, A: Sized, B> PartialEq<[A; 32]> for &'b [B]                [src]
where
    B: PartialEq<A>,

impl<'a, 'b, A: Sized, B> PartialEq<&'b mut [B]> for [A; 32]            [src]
where
    A: PartialEq<B>,

impl<'a, 'b, A: Sized, B> PartialEq<[A; 32]> for &'b mut [B]            [src]
where
    B: PartialEq<A>,

impl<T: PartialEq + Copy> PartialEq for Cell<T>                         [src]

impl<T: ?Sized + PartialEq> PartialEq for RefCell<T>                    [src]

impl PartialEq for ParseCharError                                       [src]

impl PartialEq for CharTryFromError                                     [src]

impl PartialEq for InvalidSequence                                      [src]

impl<T: PartialEq> PartialEq for Option<T>                              [src]

impl PartialEq for NoneError                                            [src]

impl<T: PartialEq, E: PartialEq> PartialEq for Result<T, E>             [src]

impl<A, B> PartialEq<[B]> for [A]                                       [src]
where
    A: PartialEq<B>,
impl PartialEq for SearchStep

impl PartialEq for str                                                    [src]

impl PartialEq for ParseBoolError                                         [src]

impl PartialEq for Utf8Error                                              [src]

impl PartialEq for Error                                                  [src]

impl PartialEq for Duration                                               [src]

impl PartialEq for Layout                                                 [src]

impl PartialEq for AllocErr                                               [src]

impl PartialEq for CannotReallocInPlace                                   [src]

impl PartialEq for CollectionAllocErr                                     [src]

impl<A> PartialEq for (A,)                                                [src]
where
    A: PartialEq + ?Sized,

impl<A: PartialEq, B> PartialEq for (A, B)                                [src]
where
    B: PartialEq + ?Sized,

impl<A: PartialEq, B: PartialEq, C> PartialEq for (A, B, C)               [src]
where
    C: PartialEq + ?Sized,

impl<A: PartialEq, B: PartialEq, C: PartialEq, D> PartialEq for (A, B, C, D)    [src]
where
    D: PartialEq + ?Sized,

impl<A: PartialEq, B: PartialEq, C: PartialEq, D: PartialEq, E> PartialEq for (A, [src]
B, C, D, E)
where
    E: PartialEq + ?Sized,

impl<A: PartialEq, B: PartialEq, C: PartialEq, D: PartialEq, E: PartialEq, F>   [src]
PartialEq for (A, B, C, D, E, F)
where
    F: PartialEq + ?Sized,

impl<A: PartialEq, B: PartialEq, C: PartialEq, D: PartialEq, E: PartialEq,      [src]
F: PartialEq, G> PartialEq for (A, B, C, D, E, F, G)
where
    G: PartialEq + ?Sized,

impl<A: PartialEq, B: PartialEq, C: PartialEq, D: PartialEq, E: PartialEq,      [src]
F: PartialEq, G: PartialEq, H> PartialEq for (A, B, C, D, E, F, G, H)
where
    H: PartialEq + ?Sized,

impl<A: PartialEq, B: PartialEq, C: PartialEq, D: PartialEq, E: PartialEq,      [src]
F: PartialEq, G: PartialEq, H: PartialEq, I> PartialEq for (A, B, C, D, E, F, G,
H, I)
where
    I: PartialEq + ?Sized,

impl<A: PartialEq, B: PartialEq, C: PartialEq, D: PartialEq, E: PartialEq,      [src]
F: PartialEq, G: PartialEq, H: PartialEq, I: PartialEq, J> PartialEq for (A, B,
C, D, E, F, G, H, I, J)
where
    J: PartialEq + ?Sized,

impl<A: PartialEq, B: PartialEq, C: PartialEq, D: PartialEq, E: PartialEq,      [src]
F: PartialEq, G: PartialEq, H: PartialEq, I: PartialEq, J: PartialEq, K>
PartialEq for (A, B, C, D, E, F, G, H, I, J, K)
where
    K: PartialEq + ?Sized,

impl<A: PartialEq, B: PartialEq, C: PartialEq, D: PartialEq, E: PartialEq,      [src]
F: PartialEq, G: PartialEq, H: PartialEq, I: PartialEq, J: PartialEq,
K: PartialEq, L> PartialEq for (A, B, C, D, E, F, G, H, I, J, K, L)
```

```
where
    L: PartialEq + ?Sized,
```

`impl PartialEq<i8x2> for i8x2` [src]

`impl PartialEq<u8x2> for u8x2` [src]

`impl PartialEq<b8x2> for b8x2` [src]

`impl PartialEq<i16x2> for i16x2` [src]

`impl PartialEq<u16x2> for u16x2` [src]

`impl PartialEq<i8x4> for i8x4` [src]

`impl PartialEq<u8x4> for u8x4` [src]

`impl PartialEq<b8x4> for b8x4` [src]

`impl PartialEq<i8x8> for i8x8` [src]

`impl PartialEq<u8x8> for u8x8` [src]

`impl PartialEq<b8x8> for b8x8` [src]

`impl PartialEq<i16x4> for i16x4` [src]

`impl PartialEq<u16x4> for u16x4` [src]

`impl PartialEq<i32x2> for i32x2` [src]

`impl PartialEq<u32x2> for u32x2` [src]

`impl PartialEq<f32x2> for f32x2` [src]

`impl PartialEq<i8x16> for i8x16` [src]

`impl PartialEq<u8x16> for u8x16` [src]

`impl PartialEq<b8x16> for b8x16` [src]

`impl PartialEq<i16x8> for i16x8` [src]

`impl PartialEq<u16x8> for u16x8` [src]

`impl PartialEq<i32x4> for i32x4` [src]

`impl PartialEq<u32x4> for u32x4` [src]

`impl PartialEq<f32x4> for f32x4` [src]

`impl PartialEq<i64x2> for i64x2` [src]

`impl PartialEq<u64x2> for u64x2` [src]

`impl PartialEq<f64x2> for f64x2` [src]

`impl PartialEq<i8x32> for i8x32` [src]

`impl PartialEq<u8x32> for u8x32` [src]

`impl PartialEq<b8x32> for b8x32` [src]

`impl PartialEq<i16x16> for i16x16` [src]

`impl PartialEq<u16x16> for u16x16` [src]

`impl PartialEq<i32x8> for i32x8` [src]

`impl PartialEq<u32x8> for u32x8` [src]

`impl PartialEq<f32x8> for f32x8` [src]

`impl PartialEq<i64x4> for i64x4` [src]

```
impl PartialEq<u64x4> for u64x4                                      [src]

impl PartialEq<f64x4> for f64x4                                      [src]

impl PartialEq<i8x64> for i8x64                                      [src]

impl PartialEq<u8x64> for u8x64                                      [src]

impl PartialEq<b8x64> for b8x64                                      [src]

impl PartialEq<i16x32> for i16x32                                    [src]

impl PartialEq<u16x32> for u16x32                                    [src]

impl PartialEq<i32x16> for i32x16                                    [src]

impl PartialEq<u32x16> for u32x16                                    [src]

impl PartialEq<f32x16> for f32x16                                    [src]

impl PartialEq<i64x8> for i64x8                                      [src]

impl PartialEq<u64x8> for u64x8                                      [src]

impl PartialEq<f64x8> for f64x8                                      [src]

impl PartialEq for CpuidResult                                       [src]

impl<T: ?Sized + PartialEq> PartialEq for Box<T>

impl<T: ?Sized + PartialEq> PartialEq for Arc<T>

impl<T: ?Sized + PartialEq> PartialEq for Rc<T>

impl<K: PartialEq, V: PartialEq> PartialEq for BTreeMap<K, V>

impl<T: PartialEq> PartialEq for BTreeSet<T>

impl<'a, 'b, B: ?Sized, C: ?Sized> PartialEq<Cow<'b, C>> for Cow<'a, B>
where
    B: PartialEq<C> + ToOwned,
    C: ToOwned,

impl<T: PartialEq> PartialEq for LinkedList<T>

impl PartialEq for String

impl<'a, 'b> PartialEq<str> for String

impl<'a, 'b> PartialEq<String> for str

impl<'a, 'b> PartialEq<&'a str> for String

impl<'a, 'b> PartialEq<String> for &'a str

impl<'a, 'b> PartialEq<str> for Cow<'a, str>

impl<'a, 'b> PartialEq<Cow<'a, str>> for str

impl<'a, 'b> PartialEq<&'b str> for Cow<'a, str>

impl<'a, 'b> PartialEq<Cow<'a, str>> for &'b str

impl<'a, 'b> PartialEq<String> for Cow<'a, str>

impl<'a, 'b> PartialEq<Cow<'a, str>> for String

impl PartialEq for ParseError

impl<'a, 'b, A: Sized, B> PartialEq<Vec<B>> for Vec<A>
where
    A: PartialEq<B>,

impl<'a, 'b, A: Sized, B> PartialEq<&'b [B]> for Vec<A>
where
    A: PartialEq<B>,
impl<'a, 'b, A: Sized, B> PartialEq<&'b mut [B]> for Vec<A>
where
    A: PartialEq<B>,

impl<'a, 'b, A: Clone, B> PartialEq<&'b [B]> for Cow<'a, [A]>
where
    A: PartialEq<B>,

impl<'a, 'b, A: Clone, B> PartialEq<&'b mut [B]> for Cow<'a, [A]>
where
    A: PartialEq<B>,

impl<'a, 'b, A: Clone, B> PartialEq<Vec<B>> for Cow<'a, [A]>
where
    A: PartialEq<B>,

impl<'a, 'b, A: Sized, B> PartialEq<[B; 0]> for Vec<A>
where
    A: PartialEq<B>,

impl<'a, 'b, A: Sized, B> PartialEq<&'b [B; 0]> for Vec<A>
where
    A: PartialEq<B>,

impl<'a, 'b, A: Sized, B> PartialEq<[B; 1]> for Vec<A>
where
    A: PartialEq<B>,

impl<'a, 'b, A: Sized, B> PartialEq<&'b [B; 1]> for Vec<A>
where
    A: PartialEq<B>,

impl<'a, 'b, A: Sized, B> PartialEq<[B; 2]> for Vec<A>
where
    A: PartialEq<B>,

impl<'a, 'b, A: Sized, B> PartialEq<&'b [B; 2]> for Vec<A>
where
    A: PartialEq<B>,

impl<'a, 'b, A: Sized, B> PartialEq<[B; 3]> for Vec<A>
where
    A: PartialEq<B>,

impl<'a, 'b, A: Sized, B> PartialEq<&'b [B; 3]> for Vec<A>
where
    A: PartialEq<B>,

impl<'a, 'b, A: Sized, B> PartialEq<[B; 4]> for Vec<A>
where
    A: PartialEq<B>,

impl<'a, 'b, A: Sized, B> PartialEq<&'b [B; 4]> for Vec<A>
where
    A: PartialEq<B>,

impl<'a, 'b, A: Sized, B> PartialEq<[B; 5]> for Vec<A>
where
    A: PartialEq<B>,

impl<'a, 'b, A: Sized, B> PartialEq<&'b [B; 5]> for Vec<A>
where
    A: PartialEq<B>,

impl<'a, 'b, A: Sized, B> PartialEq<[B; 6]> for Vec<A>
where
    A: PartialEq<B>,

impl<'a, 'b, A: Sized, B> PartialEq<&'b [B; 6]> for Vec<A>
where
    A: PartialEq<B>,

impl<'a, 'b, A: Sized, B> PartialEq<[B; 7]> for Vec<A>
where
    A: PartialEq<B>,

impl<'a, 'b, A: Sized, B> PartialEq<&'b [B; 7]> for Vec<A>
where
    A: PartialEq<B>,

impl<'a, 'b, A: Sized, B> PartialEq<[B; 8]> for Vec<A>
where
    A: PartialEq<B>,
impl<'a, 'b, A: Sized, B> PartialEq<&'b [B; 8]> for Vec<A>
where
    A: PartialEq<B>,

impl<'a, 'b, A: Sized, B> PartialEq<[B; 9]> for Vec<A>
where
    A: PartialEq<B>,

impl<'a, 'b, A: Sized, B> PartialEq<&'b [B; 9]> for Vec<A>
where
    A: PartialEq<B>,

impl<'a, 'b, A: Sized, B> PartialEq<[B; 10]> for Vec<A>
where
    A: PartialEq<B>,

impl<'a, 'b, A: Sized, B> PartialEq<&'b [B; 10]> for Vec<A>
where
    A: PartialEq<B>,

impl<'a, 'b, A: Sized, B> PartialEq<[B; 11]> for Vec<A>
where
    A: PartialEq<B>,

impl<'a, 'b, A: Sized, B> PartialEq<&'b [B; 11]> for Vec<A>
where
    A: PartialEq<B>,

impl<'a, 'b, A: Sized, B> PartialEq<[B; 12]> for Vec<A>
where
    A: PartialEq<B>,

impl<'a, 'b, A: Sized, B> PartialEq<&'b [B; 12]> for Vec<A>
where
    A: PartialEq<B>,

impl<'a, 'b, A: Sized, B> PartialEq<[B; 13]> for Vec<A>
where
    A: PartialEq<B>,

impl<'a, 'b, A: Sized, B> PartialEq<&'b [B; 13]> for Vec<A>
where
    A: PartialEq<B>,

impl<'a, 'b, A: Sized, B> PartialEq<[B; 14]> for Vec<A>
where
    A: PartialEq<B>,

impl<'a, 'b, A: Sized, B> PartialEq<&'b [B; 14]> for Vec<A>
where
    A: PartialEq<B>,

impl<'a, 'b, A: Sized, B> PartialEq<[B; 15]> for Vec<A>
where
    A: PartialEq<B>,

impl<'a, 'b, A: Sized, B> PartialEq<&'b [B; 15]> for Vec<A>
where
    A: PartialEq<B>,

impl<'a, 'b, A: Sized, B> PartialEq<[B; 16]> for Vec<A>
where
    A: PartialEq<B>,

impl<'a, 'b, A: Sized, B> PartialEq<&'b [B; 16]> for Vec<A>
where
    A: PartialEq<B>,

impl<'a, 'b, A: Sized, B> PartialEq<[B; 17]> for Vec<A>
where
    A: PartialEq<B>,

impl<'a, 'b, A: Sized, B> PartialEq<&'b [B; 17]> for Vec<A>
where
    A: PartialEq<B>,

impl<'a, 'b, A: Sized, B> PartialEq<[B; 18]> for Vec<A>
where
    A: PartialEq<B>,

impl<'a, 'b, A: Sized, B> PartialEq<&'b [B; 18]> for Vec<A>
where
    A: PartialEq<B>,
impl<'a, 'b, A: Sized, B> PartialEq<[B; 19]> for Vec<A>
where
    A: PartialEq<B>,

impl<'a, 'b, A: Sized, B> PartialEq<&'b [B; 19]> for Vec<A>
where
    A: PartialEq<B>,

impl<'a, 'b, A: Sized, B> PartialEq<[B; 20]> for Vec<A>
where
    A: PartialEq<B>,

impl<'a, 'b, A: Sized, B> PartialEq<&'b [B; 20]> for Vec<A>
where
    A: PartialEq<B>,

impl<'a, 'b, A: Sized, B> PartialEq<[B; 21]> for Vec<A>
where
    A: PartialEq<B>,

impl<'a, 'b, A: Sized, B> PartialEq<&'b [B; 21]> for Vec<A>
where
    A: PartialEq<B>,

impl<'a, 'b, A: Sized, B> PartialEq<[B; 22]> for Vec<A>
where
    A: PartialEq<B>,

impl<'a, 'b, A: Sized, B> PartialEq<&'b [B; 22]> for Vec<A>
where
    A: PartialEq<B>,

impl<'a, 'b, A: Sized, B> PartialEq<[B; 23]> for Vec<A>
where
    A: PartialEq<B>,

impl<'a, 'b, A: Sized, B> PartialEq<&'b [B; 23]> for Vec<A>
where
    A: PartialEq<B>,

impl<'a, 'b, A: Sized, B> PartialEq<[B; 24]> for Vec<A>
where
    A: PartialEq<B>,

impl<'a, 'b, A: Sized, B> PartialEq<&'b [B; 24]> for Vec<A>
where
    A: PartialEq<B>,

impl<'a, 'b, A: Sized, B> PartialEq<[B; 25]> for Vec<A>
where
    A: PartialEq<B>,

impl<'a, 'b, A: Sized, B> PartialEq<&'b [B; 25]> for Vec<A>
where
    A: PartialEq<B>,

impl<'a, 'b, A: Sized, B> PartialEq<[B; 26]> for Vec<A>
where
    A: PartialEq<B>,

impl<'a, 'b, A: Sized, B> PartialEq<&'b [B; 26]> for Vec<A>
where
    A: PartialEq<B>,

impl<'a, 'b, A: Sized, B> PartialEq<[B; 27]> for Vec<A>
where
    A: PartialEq<B>,

impl<'a, 'b, A: Sized, B> PartialEq<&'b [B; 27]> for Vec<A>
where
    A: PartialEq<B>,

impl<'a, 'b, A: Sized, B> PartialEq<[B; 28]> for Vec<A>
where
    A: PartialEq<B>,

impl<'a, 'b, A: Sized, B> PartialEq<&'b [B; 28]> for Vec<A>
where
    A: PartialEq<B>,

impl<'a, 'b, A: Sized, B> PartialEq<[B; 29]> for Vec<A>
where
    A: PartialEq<B>,
impl<'a, 'b, A: Sized, B> PartialEq<&'b [B; 29]> for Vec<A>
where
    A: PartialEq<B>,

impl<'a, 'b, A: Sized, B> PartialEq<[B; 30]> for Vec<A>
where
    A: PartialEq<B>,

impl<'a, 'b, A: Sized, B> PartialEq<&'b [B; 30]> for Vec<A>
where
    A: PartialEq<B>,

impl<'a, 'b, A: Sized, B> PartialEq<[B; 31]> for Vec<A>
where
    A: PartialEq<B>,

impl<'a, 'b, A: Sized, B> PartialEq<&'b [B; 31]> for Vec<A>
where
    A: PartialEq<B>,

impl<'a, 'b, A: Sized, B> PartialEq<[B; 32]> for Vec<A>
where
    A: PartialEq<B>,

impl<'a, 'b, A: Sized, B> PartialEq<&'b [B; 32]> for Vec<A>
where
    A: PartialEq<B>,

impl<A: PartialEq> PartialEq for VecDeque<A>

impl<'a, 'b, A: Sized, B> PartialEq<Vec<B>> for VecDeque<A>
where
    A: PartialEq<B>,

impl<'a, 'b, A: Sized, B> PartialEq<&'b [B]> for VecDeque<A>
where
    A: PartialEq<B>,

impl<'a, 'b, A: Sized, B> PartialEq<&'b mut [B]> for VecDeque<A>
where
    A: PartialEq<B>,

impl<'a, 'b, A: Sized, B> PartialEq<[B; 0]> for VecDeque<A>
where
    A: PartialEq<B>,

impl<'a, 'b, A: Sized, B> PartialEq<&'b [B; 0]> for VecDeque<A>
where
    A: PartialEq<B>,

impl<'a, 'b, A: Sized, B> PartialEq<&'b mut [B; 0]> for VecDeque<A>
where
    A: PartialEq<B>,

impl<'a, 'b, A: Sized, B> PartialEq<[B; 1]> for VecDeque<A>
where
    A: PartialEq<B>,

impl<'a, 'b, A: Sized, B> PartialEq<&'b [B; 1]> for VecDeque<A>
where
    A: PartialEq<B>,

impl<'a, 'b, A: Sized, B> PartialEq<&'b mut [B; 1]> for VecDeque<A>
where
    A: PartialEq<B>,

impl<'a, 'b, A: Sized, B> PartialEq<[B; 2]> for VecDeque<A>
where
    A: PartialEq<B>,

impl<'a, 'b, A: Sized, B> PartialEq<&'b [B; 2]> for VecDeque<A>
where
    A: PartialEq<B>,

impl<'a, 'b, A: Sized, B> PartialEq<&'b mut [B; 2]> for VecDeque<A>
where
    A: PartialEq<B>,

impl<'a, 'b, A: Sized, B> PartialEq<[B; 3]> for VecDeque<A>
where
    A: PartialEq<B>,

impl<'a, 'b, A: Sized, B> PartialEq<&'b [B; 3]> for VecDeque<A>
```

```
where
    A: PartialEq<B>,

impl<'a, 'b, A: Sized, B> PartialEq<&'b mut [B; 3]> for VecDeque<A>
where
    A: PartialEq<B>,

impl<'a, 'b, A: Sized, B> PartialEq<[B; 4]> for VecDeque<A>
where
    A: PartialEq<B>,

impl<'a, 'b, A: Sized, B> PartialEq<&'b [B; 4]> for VecDeque<A>
where
    A: PartialEq<B>,

impl<'a, 'b, A: Sized, B> PartialEq<&'b mut [B; 4]> for VecDeque<A>
where
    A: PartialEq<B>,

impl<'a, 'b, A: Sized, B> PartialEq<[B; 5]> for VecDeque<A>
where
    A: PartialEq<B>,

impl<'a, 'b, A: Sized, B> PartialEq<&'b [B; 5]> for VecDeque<A>
where
    A: PartialEq<B>,

impl<'a, 'b, A: Sized, B> PartialEq<&'b mut [B; 5]> for VecDeque<A>
where
    A: PartialEq<B>,

impl<'a, 'b, A: Sized, B> PartialEq<[B; 6]> for VecDeque<A>
where
    A: PartialEq<B>,

impl<'a, 'b, A: Sized, B> PartialEq<&'b [B; 6]> for VecDeque<A>
where
    A: PartialEq<B>,

impl<'a, 'b, A: Sized, B> PartialEq<&'b mut [B; 6]> for VecDeque<A>
where
    A: PartialEq<B>,

impl<'a, 'b, A: Sized, B> PartialEq<[B; 7]> for VecDeque<A>
where
    A: PartialEq<B>,

impl<'a, 'b, A: Sized, B> PartialEq<&'b [B; 7]> for VecDeque<A>
where
    A: PartialEq<B>,

impl<'a, 'b, A: Sized, B> PartialEq<&'b mut [B; 7]> for VecDeque<A>
where
    A: PartialEq<B>,

impl<'a, 'b, A: Sized, B> PartialEq<[B; 8]> for VecDeque<A>
where
    A: PartialEq<B>,

impl<'a, 'b, A: Sized, B> PartialEq<&'b [B; 8]> for VecDeque<A>
where
    A: PartialEq<B>,

impl<'a, 'b, A: Sized, B> PartialEq<&'b mut [B; 8]> for VecDeque<A>
where
    A: PartialEq<B>,

impl<'a, 'b, A: Sized, B> PartialEq<[B; 9]> for VecDeque<A>
where
    A: PartialEq<B>,

impl<'a, 'b, A: Sized, B> PartialEq<&'b [B; 9]> for VecDeque<A>
where
    A: PartialEq<B>,

impl<'a, 'b, A: Sized, B> PartialEq<&'b mut [B; 9]> for VecDeque<A>
where
    A: PartialEq<B>,

impl<'a, 'b, A: Sized, B> PartialEq<[B; 10]> for VecDeque<A>
where
    A: PartialEq<B>,
impl<'a, 'b, A: Sized, B> PartialEq<&'b [B; 10]> for VecDeque<A>
where
    A: PartialEq<B>,

impl<'a, 'b, A: Sized, B> PartialEq<&'b mut [B; 10]> for VecDeque<A>
where
    A: PartialEq<B>,

impl<'a, 'b, A: Sized, B> PartialEq<[B; 11]> for VecDeque<A>
where
    A: PartialEq<B>,

impl<'a, 'b, A: Sized, B> PartialEq<&'b [B; 11]> for VecDeque<A>
where
    A: PartialEq<B>,

impl<'a, 'b, A: Sized, B> PartialEq<&'b mut [B; 11]> for VecDeque<A>
where
    A: PartialEq<B>,

impl<'a, 'b, A: Sized, B> PartialEq<[B; 12]> for VecDeque<A>
where
    A: PartialEq<B>,

impl<'a, 'b, A: Sized, B> PartialEq<&'b [B; 12]> for VecDeque<A>
where
    A: PartialEq<B>,

impl<'a, 'b, A: Sized, B> PartialEq<&'b mut [B; 12]> for VecDeque<A>
where
    A: PartialEq<B>,

impl<'a, 'b, A: Sized, B> PartialEq<[B; 13]> for VecDeque<A>
where
    A: PartialEq<B>,

impl<'a, 'b, A: Sized, B> PartialEq<&'b [B; 13]> for VecDeque<A>
where
    A: PartialEq<B>,

impl<'a, 'b, A: Sized, B> PartialEq<&'b mut [B; 13]> for VecDeque<A>
where
    A: PartialEq<B>,

impl<'a, 'b, A: Sized, B> PartialEq<[B; 14]> for VecDeque<A>
where
    A: PartialEq<B>,

impl<'a, 'b, A: Sized, B> PartialEq<&'b [B; 14]> for VecDeque<A>
where
    A: PartialEq<B>,

impl<'a, 'b, A: Sized, B> PartialEq<&'b mut [B; 14]> for VecDeque<A>
where
    A: PartialEq<B>,

impl<'a, 'b, A: Sized, B> PartialEq<[B; 15]> for VecDeque<A>
where
    A: PartialEq<B>,

impl<'a, 'b, A: Sized, B> PartialEq<&'b [B; 15]> for VecDeque<A>
where
    A: PartialEq<B>,

impl<'a, 'b, A: Sized, B> PartialEq<&'b mut [B; 15]> for VecDeque<A>
where
    A: PartialEq<B>,

impl<'a, 'b, A: Sized, B> PartialEq<[B; 16]> for VecDeque<A>
where
    A: PartialEq<B>,

impl<'a, 'b, A: Sized, B> PartialEq<&'b [B; 16]> for VecDeque<A>
where
    A: PartialEq<B>,

impl<'a, 'b, A: Sized, B> PartialEq<&'b mut [B; 16]> for VecDeque<A>
where
    A: PartialEq<B>,

impl<'a, 'b, A: Sized, B> PartialEq<[B; 17]> for VecDeque<A>
where
    A: PartialEq<B>,
impl<'a, 'b, A: Sized, B> PartialEq<&'b [B; 17]> for VecDeque<A>
where
    A: PartialEq<B>,

impl<'a, 'b, A: Sized, B> PartialEq<&'b mut [B; 17]> for VecDeque<A>
where
    A: PartialEq<B>,

impl<'a, 'b, A: Sized, B> PartialEq<[B; 18]> for VecDeque<A>
where
    A: PartialEq<B>,

impl<'a, 'b, A: Sized, B> PartialEq<&'b [B; 18]> for VecDeque<A>
where
    A: PartialEq<B>,

impl<'a, 'b, A: Sized, B> PartialEq<&'b mut [B; 18]> for VecDeque<A>
where
    A: PartialEq<B>,

impl<'a, 'b, A: Sized, B> PartialEq<[B; 19]> for VecDeque<A>
where
    A: PartialEq<B>,

impl<'a, 'b, A: Sized, B> PartialEq<&'b [B; 19]> for VecDeque<A>
where
    A: PartialEq<B>,

impl<'a, 'b, A: Sized, B> PartialEq<&'b mut [B; 19]> for VecDeque<A>
where
    A: PartialEq<B>,

impl<'a, 'b, A: Sized, B> PartialEq<[B; 20]> for VecDeque<A>
where
    A: PartialEq<B>,

impl<'a, 'b, A: Sized, B> PartialEq<&'b [B; 20]> for VecDeque<A>
where
    A: PartialEq<B>,

impl<'a, 'b, A: Sized, B> PartialEq<&'b mut [B; 20]> for VecDeque<A>
where
    A: PartialEq<B>,

impl<'a, 'b, A: Sized, B> PartialEq<[B; 21]> for VecDeque<A>
where
    A: PartialEq<B>,

impl<'a, 'b, A: Sized, B> PartialEq<&'b [B; 21]> for VecDeque<A>
where
    A: PartialEq<B>,

impl<'a, 'b, A: Sized, B> PartialEq<&'b mut [B; 21]> for VecDeque<A>
where
    A: PartialEq<B>,

impl<'a, 'b, A: Sized, B> PartialEq<[B; 22]> for VecDeque<A>
where
    A: PartialEq<B>,

impl<'a, 'b, A: Sized, B> PartialEq<&'b [B; 22]> for VecDeque<A>
where
    A: PartialEq<B>,

impl<'a, 'b, A: Sized, B> PartialEq<&'b mut [B; 22]> for VecDeque<A>
where
    A: PartialEq<B>,

impl<'a, 'b, A: Sized, B> PartialEq<[B; 23]> for VecDeque<A>
where
    A: PartialEq<B>,

impl<'a, 'b, A: Sized, B> PartialEq<&'b [B; 23]> for VecDeque<A>
where
    A: PartialEq<B>,

impl<'a, 'b, A: Sized, B> PartialEq<&'b mut [B; 23]> for VecDeque<A>
where
    A: PartialEq<B>,

impl<'a, 'b, A: Sized, B> PartialEq<[B; 24]> for VecDeque<A>
where
    A: PartialEq<B>,
impl<'a, 'b, A: Sized, B> PartialEq<&'b [B; 24]> for VecDeque<A>
where
    A: PartialEq<B>,

impl<'a, 'b, A: Sized, B> PartialEq<&'b mut [B; 24]> for VecDeque<A>
where
    A: PartialEq<B>,

impl<'a, 'b, A: Sized, B> PartialEq<[B; 25]> for VecDeque<A>
where
    A: PartialEq<B>,

impl<'a, 'b, A: Sized, B> PartialEq<&'b [B; 25]> for VecDeque<A>
where
    A: PartialEq<B>,

impl<'a, 'b, A: Sized, B> PartialEq<&'b mut [B; 25]> for VecDeque<A>
where
    A: PartialEq<B>,

impl<'a, 'b, A: Sized, B> PartialEq<[B; 26]> for VecDeque<A>
where
    A: PartialEq<B>,

impl<'a, 'b, A: Sized, B> PartialEq<&'b [B; 26]> for VecDeque<A>
where
    A: PartialEq<B>,

impl<'a, 'b, A: Sized, B> PartialEq<&'b mut [B; 26]> for VecDeque<A>
where
    A: PartialEq<B>,

impl<'a, 'b, A: Sized, B> PartialEq<[B; 27]> for VecDeque<A>
where
    A: PartialEq<B>,

impl<'a, 'b, A: Sized, B> PartialEq<&'b [B; 27]> for VecDeque<A>
where
    A: PartialEq<B>,

impl<'a, 'b, A: Sized, B> PartialEq<&'b mut [B; 27]> for VecDeque<A>
where
    A: PartialEq<B>,

impl<'a, 'b, A: Sized, B> PartialEq<[B; 28]> for VecDeque<A>
where
    A: PartialEq<B>,

impl<'a, 'b, A: Sized, B> PartialEq<&'b [B; 28]> for VecDeque<A>
where
    A: PartialEq<B>,

impl<'a, 'b, A: Sized, B> PartialEq<&'b mut [B; 28]> for VecDeque<A>
where
    A: PartialEq<B>,

impl<'a, 'b, A: Sized, B> PartialEq<[B; 29]> for VecDeque<A>
where
    A: PartialEq<B>,

impl<'a, 'b, A: Sized, B> PartialEq<&'b [B; 29]> for VecDeque<A>
where
    A: PartialEq<B>,

impl<'a, 'b, A: Sized, B> PartialEq<&'b mut [B; 29]> for VecDeque<A>
where
    A: PartialEq<B>,

impl<'a, 'b, A: Sized, B> PartialEq<[B; 30]> for VecDeque<A>
where
    A: PartialEq<B>,

impl<'a, 'b, A: Sized, B> PartialEq<&'b [B; 30]> for VecDeque<A>
where
    A: PartialEq<B>,

impl<'a, 'b, A: Sized, B> PartialEq<&'b mut [B; 30]> for VecDeque<A>
where
    A: PartialEq<B>,

impl<'a, 'b, A: Sized, B> PartialEq<[B; 31]> for VecDeque<A>
where
    A: PartialEq<B>,
impl<'a, 'b, A: Sized, B> PartialEq<&'b [B; 31]> for VecDeque<A>
where
    A: PartialEq<B>,

impl<'a, 'b, A: Sized, B> PartialEq<&'b mut [B; 31]> for VecDeque<A>
where
    A: PartialEq<B>,

impl<'a, 'b, A: Sized, B> PartialEq<[B; 32]> for VecDeque<A>
where
    A: PartialEq<B>,

impl<'a, 'b, A: Sized, B> PartialEq<&'b [B; 32]> for VecDeque<A>
where
    A: PartialEq<B>,

impl<'a, 'b, A: Sized, B> PartialEq<&'b mut [B; 32]> for VecDeque<A>
where
    A: PartialEq<B>,

impl PartialEq for Span

impl PartialEq for LineColumn

impl PartialEq for SourceFile

impl PartialEq<FileName> for SourceFile

impl PartialEq for Delimiter

impl PartialEq for Spacing

impl PartialEq for UnicodeVersion

impl PartialEq for DecodeUtf16Error

impl<'a> PartialEq for Utf8LossyChunk<'a>

impl PartialEq for Summary

impl PartialEq for TestName

impl PartialEq for NamePadding

impl PartialEq for BenchMode

impl PartialEq for ShouldPanic

impl PartialEq for TestDesc

impl PartialEq for Metric

impl PartialEq for OutputFormat

impl PartialEq for BenchSamples

impl PartialEq for TestResult

impl PartialEq for MetricMap
```