A value was moved whose size was not known at compile time.

Erroneous code example:

```
#![feature(box_syntax)]
trait Bar {
    fn f(self);
}

impl Bar for i32 {
    fn f(self) {}
}

fn main() {
    let b: Box<dyn Bar> = box (0 as i32);
    b.f();
    // error: cannot move a value of type dyn Bar: the size of dyn Bar cannot
    //        be statically determined
}
```

In Rust, you can only move a value when its size is known at compile time.

To work around this restriction, consider "hiding" the value behind a reference: either `&x` or `&mut x`. Since a reference has a fixed size, this lets you move it around as usual. Example:

```
#![feature(box_syntax)]

trait Bar {
    fn f(&self);
}

impl Bar for i32 {
    fn f(&self) {}
}

fn main() {
    let b: Box<dyn Bar> = box (0 as i32);
    b.f();
    // ok!
}
```