```rust
struct MyIter<T> {
    list: Vec<T>,  // La lista di elementi
    current: usize, // Indice dell'elemento corrente
}

impl<T> MyIter<T> {
    fn new(list: Vec<T>) -> Self {
        Self { list, current: 0 }
    }
}

impl<T> Iterator for MyIter<T> {
    type Item = T;

    fn next(&mut self) -> Option<Self::Item> {
        if self.current < self.list.len() {
            let item = self.list[self.current].clone();
            self.current += 1;
            Some(item)
        } else {
            None
        }
    }
}

fn main() {
    let my_list = vec![1, 2, 3, 4, 5];
    let mut iter = MyIter::new(my_list);

    while let Some(value) = iter.next() {
        println!("{}", value); // Stampa: 1 2 3 4 5
    }
}
```

PDF document made with CodePrint using Prism