Resizing a Vector

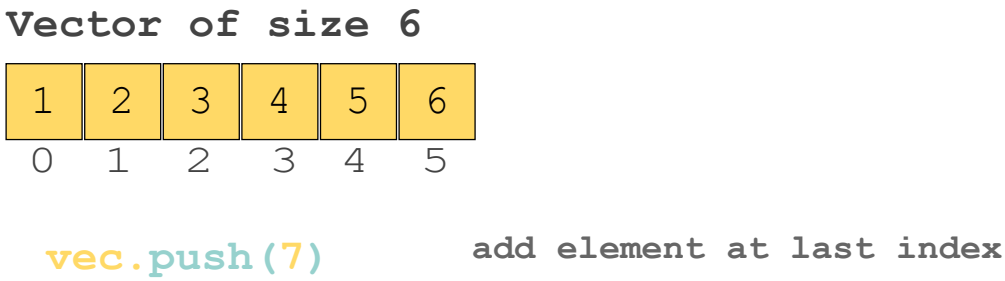
This lesson teaches how a vector can be resized or how it can grow and shrink.

We'll cover the following

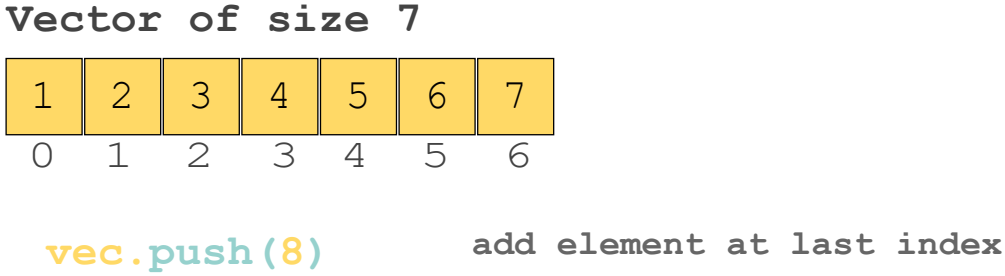
Add Elements to the Vector

- Define a mutable vector variable.
- To add elements to the vector, use the `push` method.

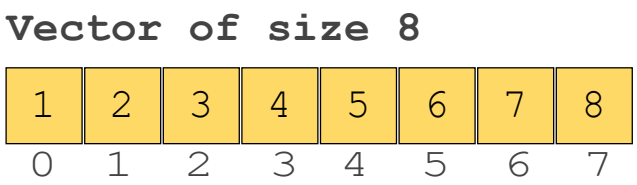
The following illustration shows how the size of the vector grows by adding an element:



1 of 3



2 of 3



3 of 3

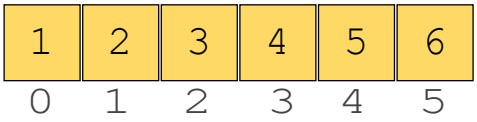
```
fn main() {
   // define a vector of size 5
   let mut my_vec = vec![1, 2, 3, 4, 5];
   // print vector
   println!("Vector : {:?}", my_vec);
   // print the capacity of vector
   println!("Capacity of vector: {}", my_vec.capacity());
   // print the length of vector
   println!("Length of the vector : {}",my_vec.len());
   my_vec.push(6);
   my_vec.push(8);
   // print vector
   println!("Vector : {:?}",my_vec);
   // print the capacity of vector
   println!("Capacity of vector: {}", my_vec.capacity());
   // print the length of vector
   println!("Length of the vector : {}", my_vec.len());
}
```

Remove Elements from the Vector

- Define a mutable vector variable.
- Elements can be removed from the tail or at specific index of the vector.
  - To remove elements from the tail of the vector, use the `pop` method.
  - To remove elements at a specific position of the vector, specify the index number within the `remove()` method.

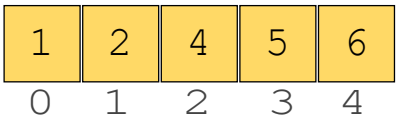
The following illustration shows how the size of the vector shrinks by removing an element:

Vector of size 6



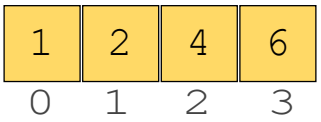
`vec.remove(2)` remove element at index 2

Vector of size 5



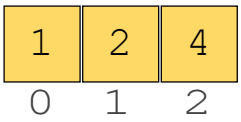
`vec.remove(3)` remove element at index 3

Vector of size 4

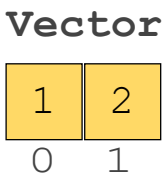


`vec.pop()` remove element at last index

Vector of size 3



`vec.pop()` remove element at last index




```
fn main() {
   // define a vector of size 5
   let mut my_vec = vec![1, 2, 3, 4, 5];
   // print vector
   println!("Vector : {:?}", my_vec);
   // print the capacity of vector
   println!("Capacity of vector: {}", my_vec.capacity());
   // print the length of vector
   println!("Length of the vector : {}", my_vec.len());
   my_vec.pop();
   my_vec.pop();
   // print vector
   println!("Vector : {:?}",my_vec);
   // print the capacity of vector
   println!("Capacity of vector: {}", my_vec.capacity());
   // print the length of vector
   println!("Length of the vector : {}", my_vec.len());
}
```

Note that the `remove()` function requires the index of the vector element to be removed. However, if it is desired to pass the element to be removed, then we need to know the index of the particular element of the vector and then remove it. Let's explore that in the next lesson using the `.iter()` method.

Quiz

Test your understanding of resizing a vector in Rust.

Quick Quiz on Resizing a Vector!

Q

What is the output of the following code?

```
fn main() {

   let mut my_vec = vec![1, 2, 3, 4, 5];
   println!("Vector : {:?}",my_vec);
   println!("Length of the vector : {}",my_vec.len());
   my_vec.push(8);
   my_vec.push(7);
   my_vec.remove(2);
   my_vec.remove(1);
   //print vector
   println!("Vector : {:?}",my_vec);
   //print the length of vector
   println!("Length of the vector : {}",my_vec.len());
}
```

Retake Quiz

Now that you have learned to resize a vector, let’s move on to the next lesson, “iterating over a vector”.