Why is an array static and the vector is dynamic?

When we declare an integer our program creates a 32 bit size space in memory and same happens when we declare an array for example when we declare an integer array of size 3 :

```
int a[3];
```

Our program creates three 32 bit spaces in memory. That means after declaration of an array we have available all the integers of the array. That means we can access any index we want after declaration.

a[0] 32bit
a[1] 32bit
a[3] 32bit

Let's say we assigned 5 in index 1.

```
int a[3];
a[1] = 5;
```

5 will be assigned in 1-no index of the array immediately as the space is created before.

a[0]
a[1] = 5
a[3]

And that's why the array is static and we have to define the size of the array before we use it.

But when we declare a vector it will have only a pointer which points the vector has ended here.

```
vector<int> v;
```

end

But when we push_back() any element in the vector it will immediately create a space for the element and assign the element in the target index.

```
vector<int> v;
v.push_back ( 3 );
```

v[0] = 3
end

For every element in the vector it will create a space first then assign the element in the target index.

```
vector<int> v;
v.push_back ( 3 );
v.push_back ( 10 );
v.push_back ( 5 );
```

v[0] = 3
v[1] = 10
v[2] = 5
end

Look vectors don't need to know the size. It increases its size by its need. That's why vectors are dynamic and don't have to declare the size. But after creating any index in vector we can access or change like a regular array, because now those indexes are available and their space has been created in memory.

And these are the reasons for why array static and vector dynamic.
Hope you enjoyed the lesson. See you in the next lesson.