1. Why ngModel Directive?
   Ans: In order to register form controls on an ngForm instance, we use the ngModel directive.
   In combination with a name attribute, ngModel creates a form control abstraction for us behind the scenes. Every form control that is registered with ngModel will automatically show up in form.value and can then easily be used for further post processing.

2. Why ngModelGroup Directive?
   Ans: ngModelGroup enables us to semantically group our form controls. In other words, there can't be a control group without controls. In addition to that, it also tracks validity state of the inner form controls.

3. Explain about ngtouched and nguntouched angular form properties?
   Ans:
   touched:- This property is used to know whether the input fields in the form are blurred / focussed or not. If the fields lost focussed then it return true and vice versa.
   Ex: formName.fieldName.touched
   untouched: This property is used to know whether the input fields in the form are blurred / focussed or not. If the fields are focussed then it returns true and vice versa.
   Ex:formName.untouched

4. Explain about Resetting Form?
   Ans: As we have done resetting in model driven forms using reset() on the myform model, but in template driven we don't have access to the form model in the component.
   So to achieve this we can use @ViewChild decorator.

5. What is @ViewChild?
   Ans: This decorator gives the reference of anything from the template in the component.
   Create a component variable with @ViewChild decorator applied to it.

6. When will we go for Custom Validators?
   Ans: When ever our validation is not possible with existing validators then we will go for custom validators.

7. what are the disadvantages of template driven forms?
   Ans: As we add more and more validator tags to a field or when we start adding complex cross-field validations the readability of the form decreases, to the point where it will be harder to hand it off to a web designer.