1. What is an expression in AngularJS?

Ans: Expressions are used to bind application data to html. Expressions are written inside double flower braces **{{expression}}.** Expressions behaves in same way as ng-bind directives. AngularJS application expressions are pure javascript expressions and outputs the data where they are used.

2. What is the difference between javascript expressions and Angular Expressions?

Ans: Angular expressions are like JavaScript expressions with the following differences:

a) Context: JavaScript expressions are evaluated against the global window. In Angular, expressions are evaluated against a scope object.
b) Forgiving: In JavaScript, trying to evaluate undefined properties generates ReferenceError or TypeError.
c) In Angular, expression evaluation is forgiving to undefined and null.
d) Filters: You can use filters within expressions to format data before displaying it.
e) No Control Flow Statements: You cannot use the following in an Angular expression conditionals, loops, or exceptions.
f) No Function Declarations: You cannot declare functions in an Angular expression, even inside ng-init directive.
g) No **Regex** Creation with Literal Notation: You cannot create regular expressions in an Angular expression.
h) No Object Creation with New Operator: You cannot use new operator in an Angular expression.
i) No Bitwise, Comma, And Void Operators: You cannot use Bitwise, , or void operators in an Angular expression.

3. What is the difference between ng-bind and Angular Expression?

Ans: The major difference between ng-bind and {{}} is that ng-bind creates a watcher for variable passed to it (i.e. name as in above example), while curly **brackets {{ }}** will (store the entire expression in memory i.e.} perform dirty-checking and refreshing the expression in every digest cycle even if it is not required.

4. Why ng-bind is better than {{}}?

Ans: It is preferable to use ngBind instead of **{{expression}}** if a template is momentarily displayed by the browser in its raw state before Angular compiles it. Since ngBind is an element attribute, it makes the bindings invisible to the user while the page is loading. **{{ }}** gives a Flashy Content because the template is loaded before AngularJS had a chance to go in and compile the elements. Whereas ng-bind is used inside HTML DOM Element.

5. What is the role of $interpolate in AngularJS?

Ans: In Angular, the $interpolate service is responsible for working with binding expression. The service returns a function you can invoke against a scope object to produce an interpolated string. The function is known as an interpolation function. Since this service is forgiving you do not see any errors, However, we can decorate the $interpolate service and wrap the interpolation functions it produces.

The wrapping function will log every execution that produces a string and warn about every interpolation that produces a false looking empty string

```
app.config(function($provide){
  $provide.decorator("$interpolate", function($delegate){
    var interpolateWrap = function(){
      var interpolationFn = $delegate.apply(this, arguments);
      if(interpolationFn) {
        return interpolationFnWrap(interpolationFn, arguments);
      }
    };
    var interpolationFnWrap = function(interpolationFn, interpolationArgs){
      return function(){
        var result = interpolationFn.apply(this, arguments);
        var log = result ? console.log : console.warn;
        log.call(console, "interpolation of  " + interpolationArgs[0].trim(),
                  ":", result.trim());
        return result;
      };
    };
    angular.extend(interpolateWrap, $delegate);
    return interpolateWrap;
  });
});
```