**Typescript Quick Notes**

Question 1) **What is the Typescript and why is used ?**

* Typescript is the super set or we can call the wrapper around the javascript where we which provides the ability to write the statically typed javascript into the code . Note the typescript is just the static protection layer , as we write the typescript code it will be converted to equivalent javascript code and then will be passed to the browser for execution.

**Question 2 )** **What is the tsc in the typescript ?**

* The tsc is the typescript compiler which is responsible to convert the typescript code to the javascript for the browser rendering .
* The **tsc compiler should be installed in the environment separately .**

**Question 3) What is the tsconfig file ?**

* This is the config fille which tells how your typescript code will be compiled to the javascript , in this file we will declare your options which will tell how our the typescript code will be compiled into the javascript .
* Some of the most important keys in the tsconfig files are :-
  + **type**:’commonjs’ or we can set whatever we need
  + **implicitany:’true’**  this will tell whether the any variable should be passed or not
  + **outputDir:’<path>’** this will tell where our javascript compiled code should be stored
  + **rootDir:’<path>’** this will be path we need to define which will be considered by the tsc compiler to compile our codes.

**To create the tsconfig file we need to use the following command tsc –init**

**Question 4) (Important) What is type inferencing the typescript ?**

* The type inferencing in typescript refers to the judging power of the typescript to find out the type of the variable during its initialization even if the type of the variable is not declared.(this is one of the most important property of the typescript code)

**Question 5) (Important) What are the Basic Types in the typescript?**

* Lets Learn about the basic types of the typescript :-
  + **number:** This type is used to denote both the integer and float variables
  + **string:**This type is used to denote string variables
  + **Boolean :** This type is used to denote the Boolean values
  + **Any :** (Dangerous to use) This is the datatype which tell that variable could have any value .
  + **Void :** (Defined for the function) This is the datatype which are used for the functions which does not return any value
  + **Null and undefined :** This datatypes will we used to declare the null or undefined if required in some cases.

**Question 6) What are some non primitive data types in the typescript?**

* Like the javascript we have the non primitive datatypes also , which are **Arrays , Tuple , enums and object**

**In this we will learn about declaring the Arrays and Tuples in the typescript :-**

**Array in the Typescript can be declared in the below syntax**

**Syntaxt1)** const arr:number[]=[1,2,3,4,5] (general syntax to declare number array)

**Syntax 2)** const arr2:Array<number>=[1,2,3,4] (using the Array generics)

**Syntax to declare the Tuples ( The tuples in the typescript) a Array with set elements with set Datatypes**

**Syntax 1)** // syntax to declare the tuples

const tup :[number,string,boolean]=[1,'Hello',true] //this is the syntax to declare the tuple in the typescript

**Question 6) What are enums in the typescript ?**

* Enums like in the other languages declares the set of constant variables which can have the set values . In the typescript we also have the concept of the enums .

To declare the enum we need to use th **enum keyword.**

Below is the syntax to declare the enum in the typescript

enum Color{

    // constant values which can be assigned

    Red,

    Green,

    Blue

}

One thing to note about the enums is that when we declare the enum values their default values will be 0 ,1 ,2 …. And so on . So to declare the enums with values below syntax is required

enum Color{

    // constant values which can be assigned

    Red=”Red”,

    Green=”Green”,

    Blue=”Blue”

}

const a:Color = Color.Green // declaring the variable with the enum

**Question 7) What are union types in the typescript ?**

* Sometime we need that the variable could have two values , like for example we need that a variable could have a value of a string or could be undefined .

**The symbol to define the union data types are (|) pipe symbol.**

Syntax to declare the union type datatype:-

// union type in typescript

let unin:string|undefined;

unin = “Hello World”

**Question 8) How to declare the objects in Typescript ( The concept of the Types)?**

* As we know that objects in the javascript is one of the most important topic , to declare the typescript object which needs to be static in nature , we use the concepts of the types

**To** Declare the type we need to **use the type keyword .**

**Below is the syntax to declare type in the typescript**

// declaring types in typescript

type User = {

    name:string,

    age:number,

    gender?:string | undefined

}

const userObj:User={

    name:"Rishabh",

    age:20,

    gender:"Male"

}

// where the ? denotes that object is optional

In types we cannot redeclare the another type with the same name

// redeclaring the type with same name is not possible

type Amount1={

    name:"Rishabh",

    age:20

}

type Amount1={

    gender:"male",

}

//Redeclaring the types is not allowed

**Question 9 ) What are type assertions in the typescript ?(Important to read more about assertions)**

* The type assertions in the typescript is the assertion we apply to the variable asserting it be of particular type . Declaring the assertion the typescript compiler will try to typecast the variable to a particular data type ( **but only if possible)**

**Below is the syntax , how to use the type assertions in the typescript:-**

Syntax 1) Using the **as keyword ( to be understand more)**

// Type assertions in the typescript

const cid:any = 20

// asserting cid to be number

let abcs= cid as number // Syntax 1)

**Functions in Typescript**

**Question 10) How to declare the functions in typescript ?**

* In the typescript while declaring the functions we also use the static type syntax
  + The Arguments should have the type declared.
  + The function return type should be declared.

Below is the syntax to declare the simple functions in the typescript

// how to declare the function in the typescript

function add (a:number,b:number):number{

    return a+b;

}

Question 11) **What are the interfaces in the typescript ?**

* The interfaces is also a way to declare the structure of the object and they are quite similar to the types.

To declare the interface we need to use the **interface keyword.**Below is the syntax to declare the interface

// Declaring the interface in the typescript

 interface Amount {

    total:number

    subTotal:number

 }

Question 12) **More About Interfaces the interface merging ?**

**🡪**Unlike the types were we cannot redeclare the types , in the interfaces we can redeclare it , and it will automatically merge the new attributes into the same interface.

**Check the Below Syntax**

// Declaring the interface in the typescript

 interface Amount {

    total:number

    subTotal:number

 }

 interface Amount{

    amountType:string

 }

 const amount:Amount={

    amountType:"Dollar",

    subTotal:20,

    total:40

 }

Question 12)**What is the difference between the type and interface Keywords?**

* The type and interface both can be used to declare the object structures in the javascript .
* But types and interface has some minute differences
  + **Differnce 1) Primitive Type allocation**
  + **Types:** can be used with the primitive data types with intersections
* // type having the primitive data type
* type edcg=number|string
  + With the interface its not possible

**Differnce2) Dynamic Merging**

**Types :** Dynamic Merging is not possible cant declare two type with the same name

**Interface :** Dynamic Merging is possible and we can declare the interface with the same name it will auto merge

**\*\*More Difference to be added**

Question 13) **What is the extends and implements keyword in typescript?**

* The extends and implements keyword is generally used with the interface to extend the interface structure .
* The extends and implements words are taken from OOP programming language where extends means the interface can take properties of the parent interface
* Implements is keywords which poses the restriction that what ever the interface we are implementing all its attributes should be used .