Q.3)Question

**a)List what you know about the differences between TS and JS and why TS is more powerful than JS?**

-1)TS:

1.typescript needs to be compiled.

2.community is small and growing.

3.Strongly typed and supports both static and dynamic typing.

4.source file extension is .ts and .tsx.

5.it takes extra time due to code compilation.

6.prior scripting knowledge is required.

7.ecosystem is powerful and intuitive.

2)JS:

1)compilation is not required.

2)community is large and evolving.

3)Loosely typed and only supports dynamic typing.

4)source file extension is .js and .jsx.

5)it enables faster coding.

6)easy to learn and understand.

-Typescript is often considered more powerful than Javascript for certain development scenarios due to its static typing feature. In typescript,you can define variable types,function parameter types and return types,providing better code clarity and catching potential errors during development.

Advantage:- 1.static typing,2.better collaboration,3.enhanced tooling.

**b) What is Generics in Ts and what is the importance of it?**

**-** Generics:- generics are a powerful tool that allows you to create reusable code that works with a variety of data types. They act like placeholders for types.

Importance of generics:-

1)building robust libraries:-generics are essential for creating well-typed and reausable libraries that can be used with various data types.

2)maintaining large codebases:- as project grow,generics help manage complexity by reducing code duplication and ensuring type safety.

3)writing cleaner and more expressive code:- generics lead to clearer and more self-documenting code,improving readability.

**c) What is the different between any and unknown and never types in TS?**

**:-1)any:-**

1)Represents any possible value:- any doesn’t restrict the type at all,allowing any value-strings, numbers,objects,functions,etc.

2)Disables type checking:-since any doesn’t provide type information,Typescript won’t catch potential type errors,sacrificing type safety.

3)Use sparingly-using any too often defeats the purpose of typescript type system.Use it only when absolutely necessary,like working with external libraries with unknown types.

**2)Unknown:**

1)Represents a value of unknown type:

Unlike any,unknown still participates in type checking,making you explicity handle type assertions or narrowing before using it.

2)More restrictive than any:- It prevents direct access to properties or methods without checks,promoting safer code compared to any.

3)Use when dealing with dynamic data:-this type is helpful when receiving data from external sources like user input or API responses where the type is uncertain.

**3)Never:-**

1)Represents a type that never occurs:-this type signifies a scenario where no value is possible.

2)Used for functions that don’t return:-Functions with return type never indicates they always throw an error or never reach the end .

3)Limited use cases:-While not very common,never helps express logic in certain situations,like defining helper function.

**d) What is the different between union and intersect types and give a small code example?**

-**Union:-**

1)Represents the combination of elements from two or more sets,including duplicates.

2)think of it as bringing elements together under one umbrella.

Symbol:- ∪

Example: A={1,2,3},B={2,4,5}

Union (A∪B)={1,2,3,4,5}

Code:-Union type

Let myVariable :string | number;

myVariable =”sardar”;

console.log(myVariable.length);// valid for string

myVariable = 23;

console.log(myVariable.toFixed(2));// valid for number

**Intersection:**

1)Contains only the elements that are common to two or more sets.

2)Think of it as the overlapping area where elements belong to both sets.

Symbol: ∩

Example:A={1,2,3},B={2,4,5}

Intersection (A∩B)={2}

Code: intersect

Type Printable ={print:()=>void};

type Loggable={ log()=>void};

type LoggableAndPrintable=Printable & Loggable;

const obj: LoggableAndPrintable={

print:()=> console.log(“Printing..”),

log: ()=>console.log(“loging..”),

};

obj.print();// accessing printable behavior

obj. log();//accessing loggable behavior

**e) TS and JS both are OOP language or not if yes or no explain the reason that makes you say that?**

- yes, both Typescript and Javascript are object-oriented programming languages.they support key OOP concepts such as classes,objects,inheritance,encapsulation,and polymorphism.Typescript,however,is a superset of javascript and introduces static typing,which allows for a more structured and robust implementation of OOP principles.

**f) What is conditional type in TS and give a line of code as example on it?**

**-**conditional types are powerful feature that allows you to define type transformation based on a condition.this enables you to create more expressive and flexible types,especially when working with generics or complex type relationships.

ConditionalType= T extends BaseType ? TrueType :FalseType;

T:the type to be checked against the condition.

BaseType:the base type that the condition checks for.

TrueType: the type to return if the condition(T extends BaseType) is true.

FalseType:the type to return if the conditions is false.

**g)How browser compiling TS?**

**-**three main ways to compile typescript for the browser:

**1)Using the Typescript Compiler(tsc):**

1)this is the official way to compile typescript.

2)you can run tsc on your typescript files to generate javascript output files.

3)this method offers fine-grained control over the compilation process and supports advanced features like source maps.

4)it requires running the compiler locally,which can be cumbersome for development workflows.

**2)Using build tools:**

1)popular build tools like webpack,rollup,or Parcel often integrate with Typescript through loaders.

2)these tools bundle your code,automatically handling typescript compilation during the build process.

3)this approach simplifies the workflow but might require configuration and understanding of the build tool.

**3)Online Transpilers:**

1)websites like the typescript playground or babel online transpiler can be used to transpile typescript to javascript directly in the browser.

**h) What is the concept of type inference?**

-1)the concept of type interface generally refers to the specifications of how different types or classes can interact with each other in a programming language.

2)it defines the set of operations that can be performed on a type and how those operations behave.

3)it specifies the properties and their types that an object must have to be considered of that particular interface type.