**JavaScript Hoisting কী? (বাংলায় ব্যাখ্যা)**

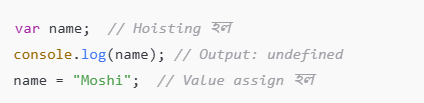
**Hoisting** হলো JavaScript-এর একটি ডিফল্ট আচরণ, যেখানে ভেরিয়েবল ও ফাংশনের ঘোষণা (declaration) স্বয়ংক্রিয়ভাবে তাদের স্কোপের শীর্ষে (top of their scope) নিয়ে যাওয়া হয় **execution** শুরুর আগে।

**ভেরিয়েবল Hoisting**

JavaScript-এ যখন কোনো ভেরিয়েবল var দিয়ে ঘোষণা করা হয়, তখন এটি স্বয়ংক্রিয়ভাবে স্কোপের শীর্ষে চলে যায়, তবে এর মান (value) **Hoist হয় না**।

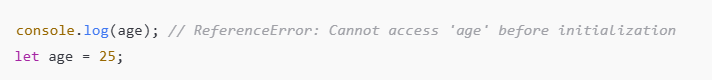
****

**কী হচ্ছে এখানে?**  
আসলে JavaScript কোডটিকে এমনভাবে রি-অ্যারেঞ্জ করে:



**let এবং const-এর ক্ষেত্রে Hoisting**

let এবং const দিয়েও ভেরিয়েবল Hoist হয়, কিন্তু এগুলো **temporal dead zone (TDZ)** এর মধ্যে থাকে, যার কারণে ব্যবহার করার আগেই এগুলো এক্সেস করলে **ReferenceError** দেখা যায়।



**ফাংশন Hoisting**

ফাংশন **Hoist হয় পুরোপুরি**, মানে এগুলোকে ডিক্লারেশনের আগে কল করা সম্ভব।



কিন্তু **function expression (var/let/const দিয়ে অ্যাসাইন করা ফাংশন)** Hoist হয় না।



এখানে শুধু hello ভেরিয়েবলটি Hoist হয়েছে, কিন্তু এটি তখনও undefined, তাই hello() কল করলে **TypeError** দেয়।

**সংক্ষেপে**

1. **var** - Hoist হয়, কিন্তু **undefined থাকে**।
2. **let & const** - Hoist হয়, কিন্তু **TDZ-এর কারণে এক্সেস করা যায় না**।
3. **Function Declaration** - পুরোপুরি Hoist হয়, আগে থেকেও কল করা যায়।
4. **Function Expression & Arrow Function** - Hoist হয় না, undefined থেকে যায়।

**পরামর্শ:**

* var এড়িয়ে let ও const ব্যবহার করা ভালো।
* ফাংশন ডিক্লারেশন সবসময় আগে লিখলে কোড পড়া সহজ হয়।

4o