|  |  |  |
| --- | --- | --- |
| SN | Question | Answer |
| 1 | Golang কেন অনেক দ্রুত কম্পাইল এবং রান করে? | Go একটি statically typed language, যার মানে হলো কম্পাইলার compile time-এ সব ধরণের টাইপ resolve করে ফেলে। এতে রানটাইমে টাইপ ইনফারেন্স বা dynamic resolution করার প্রয়োজন হয় না। এটি কোড দ্রুত কম্পাইল ও রান করতে সহায়তা করে। |
| 2 | Golang এর কয়েকটি সুবিধা উল্লেখ করুন? | ১. **সরল ও সহজবোধ্য সিনট্যাক্স**   * Go এর সিনট্যাক্স খুবই সহজ এবং CLEAN, যা পড়া ও বোঝা সহজ। এটি C-এর মতো দেখতে, কিন্তু অনেক বেশি মডার্ন এবং সংক্ষিপ্ত।   ২. **দ্রুত কম্পাইলেশন ও এক্সিকিউশন**   * Go একটি কম্পাইল্ড ভাষা, তাই এটি দ্রুত রান করে। এর কম্পাইলেশন সময়ও খুবই কম।   ৩. **কনকারেন্সি সাপোর্ট (Goroutines & Channels)**   * Go-তে **Goroutines** (হালকা থ্রেড) এবং **Channels** এর মাধ্যমে কনকারেন্সি ম্যানেজমেন্ট সহজ। এটি পারফরম্যান্স-ক্রিটিকাল অ্যাপ্লিকেশনের জন্য আদর্শ।   ৪. **গার্বেজ কালেকশন (Automatic Memory Management)**   * গার্বেজ কালেকশন (GC) থাকায় মেমোরি ম্যানেজমেন্ট অনেক সহজ, মেমোরি লিকের ঝামেলা কম।   ৫. **ক্রস-প্ল্যাটফর্ম সাপোর্ট**   * Go বিভিন্ন প্ল্যাটফর্ম (Linux, Windows, macOS) এবং আর্কিটেকচার (x86, ARM) সাপোর্ট করে। একক কোডবেইস দিয়ে বিভিন্ন প্ল্যাটফর্মে বিল্ড করা যায়।   ৬. **বিল্ট-ইন টেস্টিং ও ডকুমেন্টেশন টুলস**   * Go-তে ইউনিট টেস্টিং (go test) এবং ডকুমেন্টেশন জেনারেশন (go doc) বিল্ট-ইন আছে, যা ডেভেলপমেন্টকে সহজ করে।   ৭. **স্ট্যান্ডার্ড লাইব্রেরির সমৃদ্ধতা**   * Go-তে নেটওয়ার্কিং, ক্রিপ্টোগ্রাফি, ফাইল হ্যান্ডলিং ইত্যাদির জন্য শক্তিশালী স্ট্যান্ডার্ড লাইব্রেরি রয়েছে।   ৮. **স্ট্যাটিক টাইপিং ও টাইপ সেফটি**   * Go স্ট্যাটিক্যালি টাইপড, তাই কম্পাইল টাইমেই অনেক বাগ ধরা পড়ে, রানটাইম এরর কম হয়।   ৯. **মাইক্রোসার্ভিসেস ও ক্লাউড কম্পিউটিং এর জন্য উপযুক্ত**   * Docker, Kubernetes, Prometheus এর মতো টুলস Go-তে লেখা, যা Go-কে ক্লাউড ও ডিস্ট্রিবিউটেড সিস্টেমের জন্য জনপ্রিয় করে তুলেছে।   ১০. **কম ডিপেন্ডেন্সি ও সিঙ্গেল বাইনারি আউটপুট**   * Go প্রোগ্রামগুলোর ডিপেন্ডেন্সি কম থাকে এবং সবকিছু একটি সিঙ্গেল বাইনারি ফাইলে কম্পাইল হয়, যা ডেপ্লয়মেন্টকে সহজ করে।   এই সুবিধাগুলোর কারণে Go মডার্ন সফটওয়্যার ডেভেলপমেন্ট, বিশেষ করে ব্যাকএন্ড, ডিস্ট্রিবিউটেড সিস্টেম এবং ক্লাউড-নেটিভ অ্যাপ্লিকেশনে ব্যাপকভাবে ব্যবহৃত হয়। |
| 3 | কিভাবে মাল্টিপল টাইপের ভ্যারিয়েবল ডিক্লারেশন একই লাইনে করতে হয় ? | * একই টাইপের ভ্যারিয়েবল: var a, b, c int * ভিন্ন টাইপের ভ্যারিয়েবল: x, y := 42, "Go" * ব্লক স্টাইলে: var ( ... )   এভাবে কম কোডে একাধিক ভ্যারিয়েবল ডিক্লেয়ার করা যায়! |
| 4 | Go কি ফাংশনাল নাকি অবজেক্ট অরিয়েন্টেড ল্যাগুয়েজ? | **o** মূলত **প্রসিডিউরাল** ও **কনকারেন্ট** ল্যাঙ্গুয়েজ, তবে এতে **অবজেক্ট-ওরিয়েন্টেড (OOP)** ও **ফাংশনাল প্রোগ্রামিং** এর কিছু ফিচার আছে।  **১. অবজেক্ট-ওরিয়েন্টেড ফিচার (সীমিত)**   * **স্ট্রাকচার (Structs)** দিয়ে ডাটা মডেল করা যায়। * **ইন্টারফেস (Interfaces)** দিয়ে পলিমরফিজম। * **মেথডস** (স্ট্রাকচারের সাথে ফাংশন যুক্ত করা যায়)। * **কোন ক্লাস বা ইনহেরিটেন্স নেই** (কম্পোজিশন ব্যবহার করতে হয়)।   **২. ফাংশনাল ফিচার (সীমিত)**   * **ফার্স্ট-ক্লাস ফাংশন** (ভ্যারিয়েবলে ফাংশন অ্যাসাইন করা যায়)। * **ক্লোজার (Closures)** সাপোর্ট করে। * **হাইয়ার-অর্ডার ফাংশন** (ফাংশন রিটার্ন/প্যারামিটার হিসেবে পাস করা যায়)। |
| 5 | **Go-তে স্লাইস (Slice) কি?**  **স্লাইস** হলো Go-তে ডাইনামিক সাইজের একটি **ফ্লেক্সিবল অ্যারে-ভিত্তিক ডাটা স্ট্রাকচার**। এটি অ্যারের মতোই কাজ করে, কিন্তু সাইজ পরিবর্তন করা যায় (ডাইনামিকালি বাড়ানো/কমানো যায়) | **স্লাইস** হলো Go-তে ডাইনামিক সাইজের একটি **ফ্লেক্সিবল অ্যারে-ভিত্তিক ডাটা স্ট্রাকচার**। এটি অ্যারের মতোই কাজ করে, কিন্তু সাইজ পরিবর্তন করা যায় (ডাইনামিকালি বাড়ানো/কমানো যায়)।  স্লাইস ইন্টারনালি **৩টি জিনিস ট্র্যাক করে**:   1. **পয়েন্টার** → underlying অ্যারের স্টার্টিং এড্রেস। 2. **লেন্থ (length)** → স্লাইসের কারেন্ট সাইজ। 3. **ক্যাপাসিটি (capacity)** → ম্যাক্সিমাম সাইজ (অ্যাডিশনাল মেমোরি অ্যালোকেশন ছাড়াই |
| 6 | স্লাইস এবং অ্যাারের মধ্যে পার্থক্য কি? | **স্লাইস vs অ্যারে (Array)**   | **ফিচার** | **অ্যারে (Array)** | **স্লাইস (Slice)** | | --- | --- | --- | | **সাইজ** | ফিক্সড (কম্পাইল টাইমে নির্ধারিত) | ডাইনামিক (রানটাইমে পরিবর্তনযোগ্য) | | **ডিক্লেয়ারেশন** | var arr [5]int | var slice []int | | **পারফরম্যান্স** | ফাস্ট (মেমোরিতে কন্টিগুয়াস) | কিছুটা স্লো (অ্যারের উপর ভিত্তি করে) | | **ইউজ কেস** | যখন ফিক্সড সাইজ দরকার | যখন ডাইনামিক সাইজ দরকার | |
| 7 | Golang এ GoPATH ভ্যারিয়েবল দ্বারা কি বোঝায়? | GOPATH হলো **Go-এর একটি এনভায়রনমেন্ট ভ্যারিয়েবল**, যা Go প্রোজেক্টের **ওয়ার্কস্পেস ডিরেক্টরি (Workspace Directory)** নির্দেশ করে। এটি মূলত তিনটি ফোল্ডার স্ট্রাকচার মেইনটেইন করে:   1. src → সোর্স কোড (.go ফাইল) 2. pkg → কম্পাইল্ড প্যাকেজ (.a ফাইল) 3. bin → এক্সিকিউটেবল বাইনারি (প্রোগ্রাম) |
| 8 |  |  |
| 9 |  |  |
| 10 |  |  |
| 11 |  |  |
| 12 |  |  |
| 13 |  |  |
| 14 |  |  |
| 15 |  |  |
| 16 |  |  |