**Assignment – 2**

**Question 1: V8 vs Chakra vs SpiderMonkey (in depth from Medium-Style Sources)**

**Engine**

**V8 (by Google/Chrome & Node.js):**

* Written in C++.
* Converts JS straight to optimized machine code using two compilers: fast-first *Ignition* and advanced *TurboFan.*
* Includes super-smart garbage collector and inline caching for speed.

**Chakra (Microsoft, used in old Edge/IE):**

* Also in C++.
* Parses JS into bytecode, then profiles and JIT-compiles, often on another CPU core.

**SpiderMonkey (Mozilla/Firefox):**

* First JS engine (Netscape). Written in C++.
* Uses intermediate bytecode (through TraceMonkey, IonMonkey, WarpMonkey), not direct to machine code.
* Generally slower than V8 and Chakra in benchmarks.

**Speed Comparison**

* **V8** is fastest—optimises JS aggressively with direct compilation + inline caching + hidden classes.
* **Chakra** is next—JITs in parallel, strong profile-based optimization.
* **SpiderMonkey** is slower—it compiles via bytecode then JIT, so less direct optimization.

**Question 2: Garbage Collector Concepts**

**1) Mark-and-Sweep Algorithm**

* Works in two steps:
  + **Mark**: Finds reachable objects.
  + **Sweep**: Removes unmarked ones, freeing memory.
* Helps reclaim memory from things no longer used.

**2) Inlining**

* Compiler optimization: replaces function calls with their body for speed.

**3) Inline Caching**

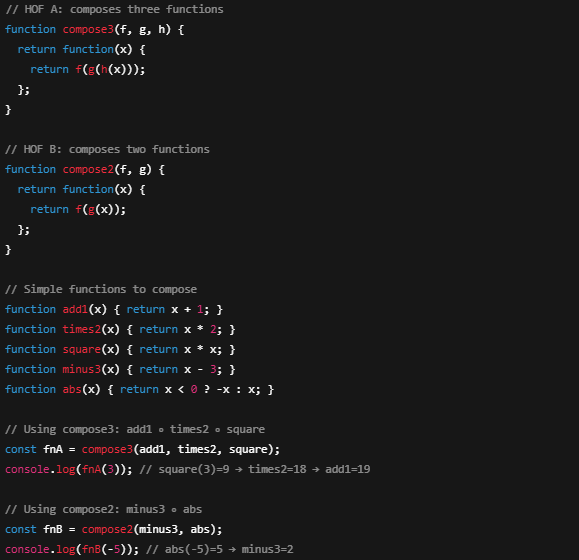
* Speeds up property access by remembering object layouts ("hidden classes").

**4) Copy Alison**

* Likely means **"copying/parallel GC"**: a collector that duplicates live objects between memory spaces, ignoring garbage.

**All four improve runtime speed and memory use when implemented in engines like V8 and Chakra.**

**Question 3: High-Order Functions & Composition**

**Code:**

**Output:**

