Output: If the code executes without errors, determine the output.
Concept: Briefly state the key programming concepts the question tests if provided in the options
Review the Code below and determine from amongst the provided options

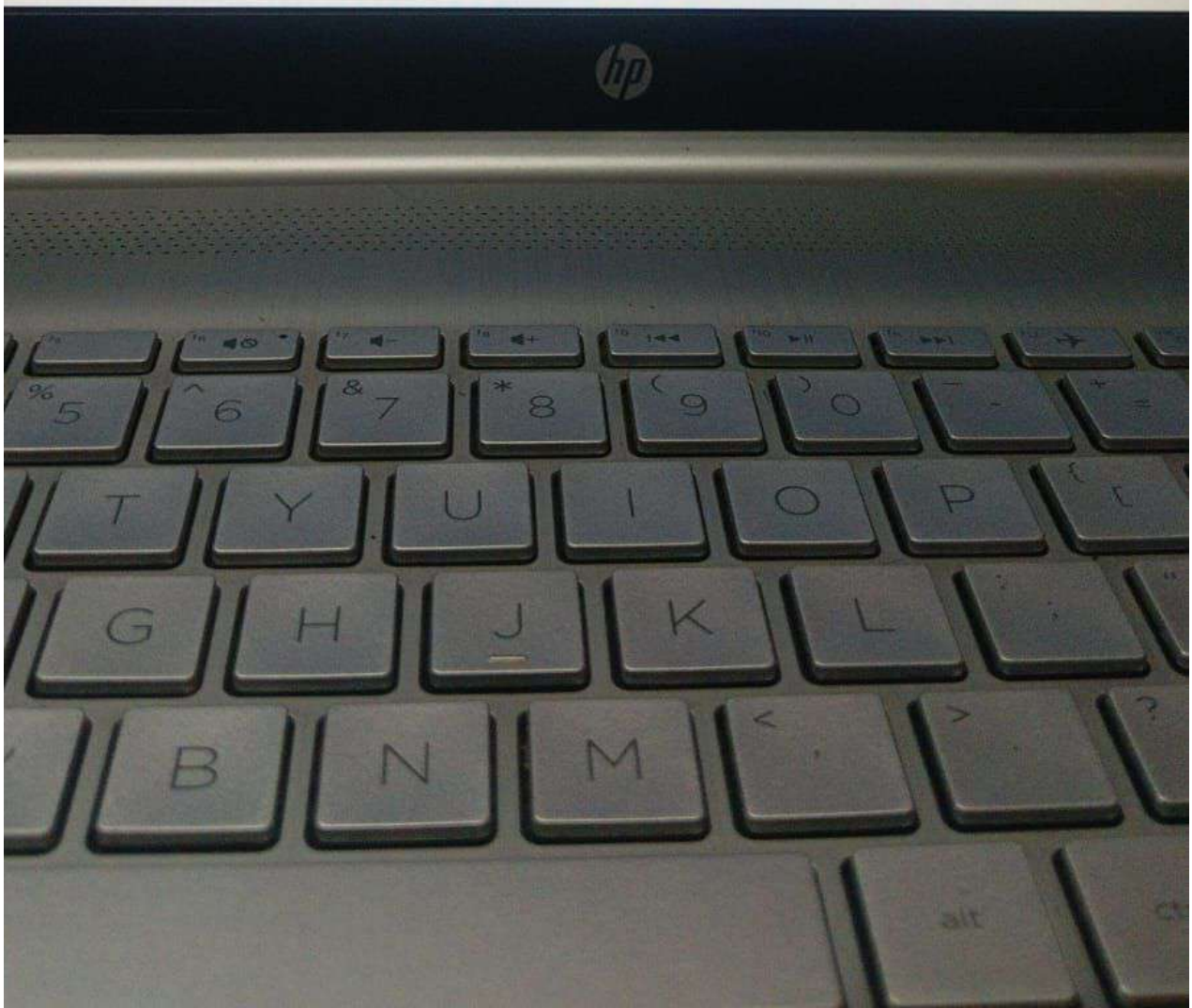
```
1 | #include <stdio.h>
2 | #include <string.h>
3 | int main() {
4 | char *str1 = "hello";
5 | char *str2 = "world";
6 | str1 = str2;
7 | printf("%s\n", str1);
8 | return 0;
9 | }
```

Answer:

- ○ The code outputs "hello"
- ○ The code outputs "world"
- ○ Compiler error: Cannot modify a string literal
- ○ Runtime Error

< Previous

Clear Answer



```
2  | void reverse_array(int *arr, int start, int end) {
3  | while (start <= end) {
4  | int temp = arr[start];
5  | arr[start] = arr[end];
6  | arr[end] = temp;
7  | start++;
8  | end--;
9  | }
10 | }
11 | int main() {
12 | int numbers[] = {1, 4, 2, 8, 5};
13 | reverse_array(numbers, 0, 4);
14 | for (int i = 0; i < 5; i++) {
15 | printf("%d ", numbers[i]);
16 | }
17 | printf("\n");
18 | return 0;
19 | }
```

Answer:

- ◯ Change the comparison in the while loop to <.
- ◯ Initialize start to 1 in main.
- ◯ The code functions correctly as intended.
- ◯ Use pointers instead of indexing in reverse_array.

< Previous

Clear Answer

```c
#include <string.h>
char* replace_substring(char *str, const char *old_str, const
char *new_str) {
char *result;
char *pos;
pos = strstr(str, old_str);
if (pos == NULL) {
return str;
}
int new_len = strlen(new_str);
int old_len = strlen(old_str);
result = (char*)malloc(strlen(str) + new_len - old_len + 1);
strncpy(result, str, pos - str);
strcpy(result + (pos - str), new_str);
strcpy(result + (pos - str) + new_len, pos + old_len);
return result;
}
```

Answer:

- ○ Change the allocation size in malloc to account for potential differences in the length of the old and new subs
- ○ Add a loop to handle multiple occurrences of old_str.
- ○ The code is perfectly correct.
- ○ Use memmove instead of strcpy for overlapping memory regions.

< Previous

Clear Answer

The C code snippets below may contain subtle errors or logic flaws. Choose the single best option to fix the code.

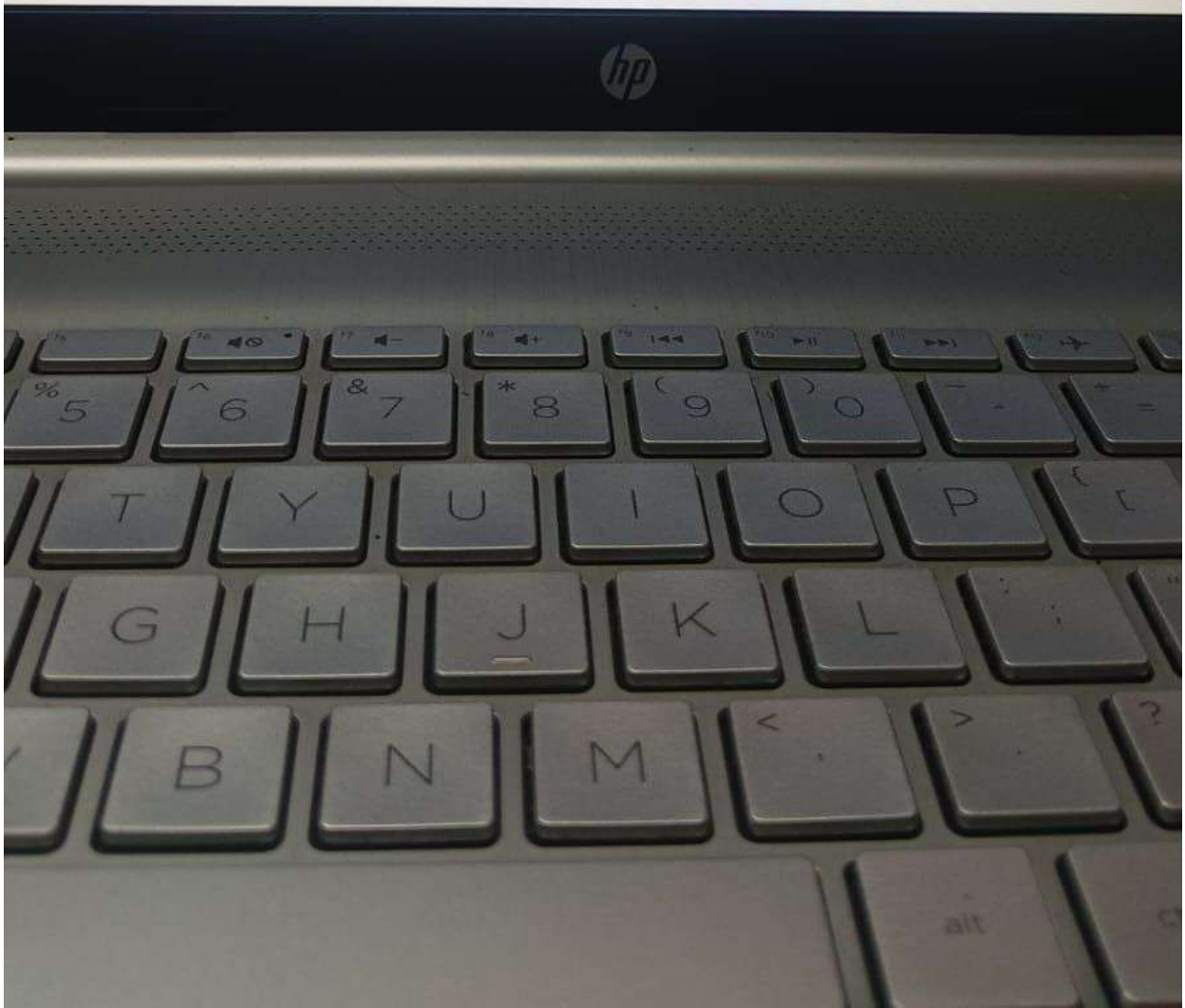
```c
#include <stdio.h>
#include <math.h>
int is_prime(int num) {
if (num <= 1) {
return 0;
}
for (int i = 2; i < num; i++) {
if (num % i == 0) {
return 0;
}
}
return 1;
}
```

Answer:

- ○ The code has no errors.
- ○ Change the initial value of i to 3.
- ○ Add a special check for the number 2.
- ○ Change the for loop condition to i <= sqrt(num).

< Previous

Clear Answer



```c
#include <stdio.h>
int find_max(int arr[], int size) {
int max = 0;
for (int i = 0; i < size; i++) {
if (arr[i] > max) {
max = arr[i];
}
}
return max;
}
int main() {
int numbers[] = {3, 7, 1, 9, 5};
int max_value = find_max(numbers, 5);
printf("Max value: %d\n", max_value);
return 0;
}
```

Answer:

- ○ Initialize max to the first element of the array: int max = arr[0];.
- ○ Change the array access in the loop to arr[i + 1].
- ○ Return a pointer to the max element within the array.
- ○ The code is perfectly correct.

< Previous

Clear Answer

What will be the output of the following Program?

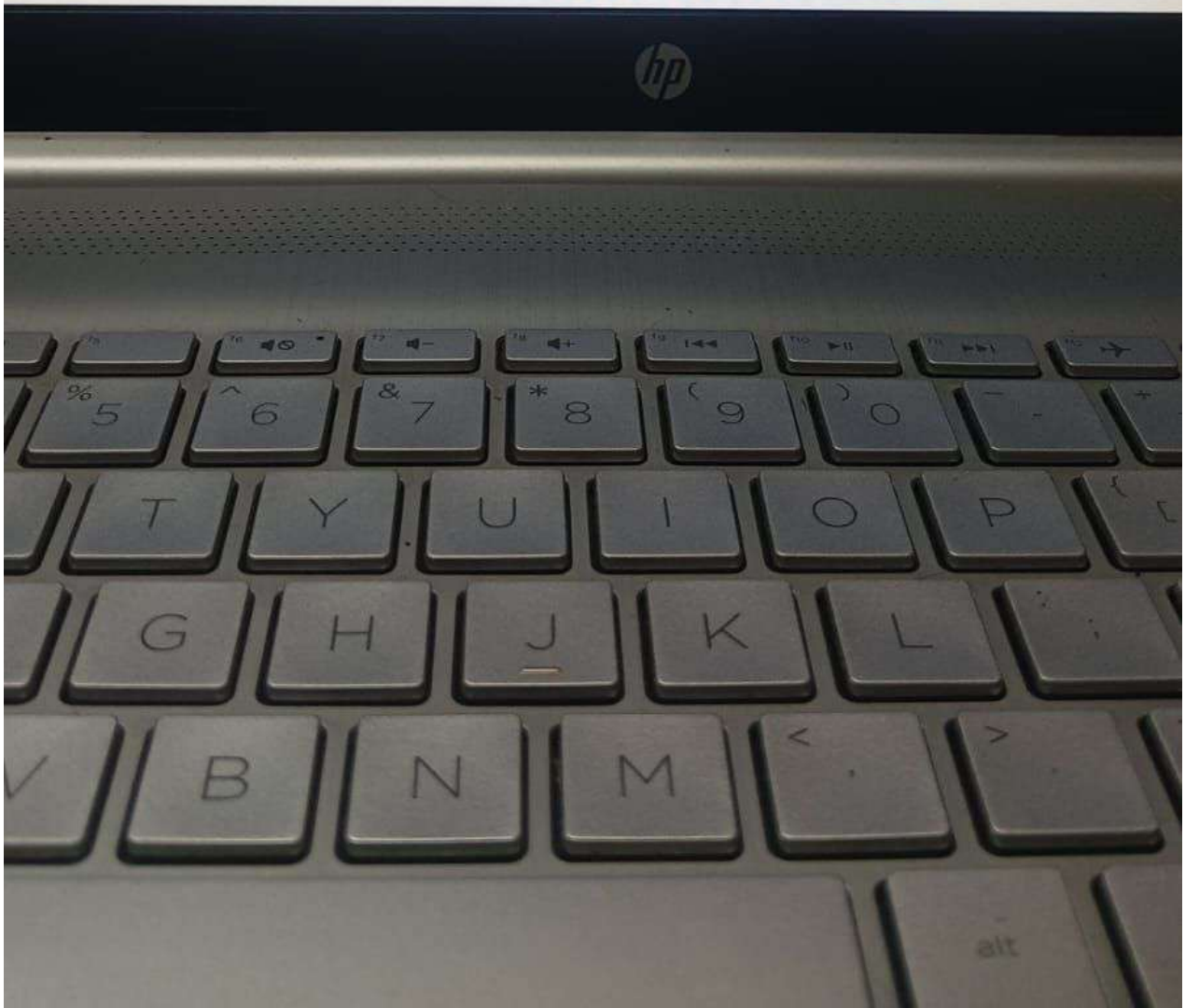
```
#include<stdio.h>
void fun(int*, int*);
int main()
{
int i=5, j=2;
fun(&i, &j);
printf("%d, %d", i, j);
return 0;
}
void fun(int *i, int *j)
{
*i = *i**i;
*j = *j**j;
}
```

Answer:

- 10, 4
- 5, 2
- 25, 4
- 2, 5

‹ Previous

Clear Answer



Concept: Briefly state the key programming concepts the question tests if provided in the options

Choose the Correct Option

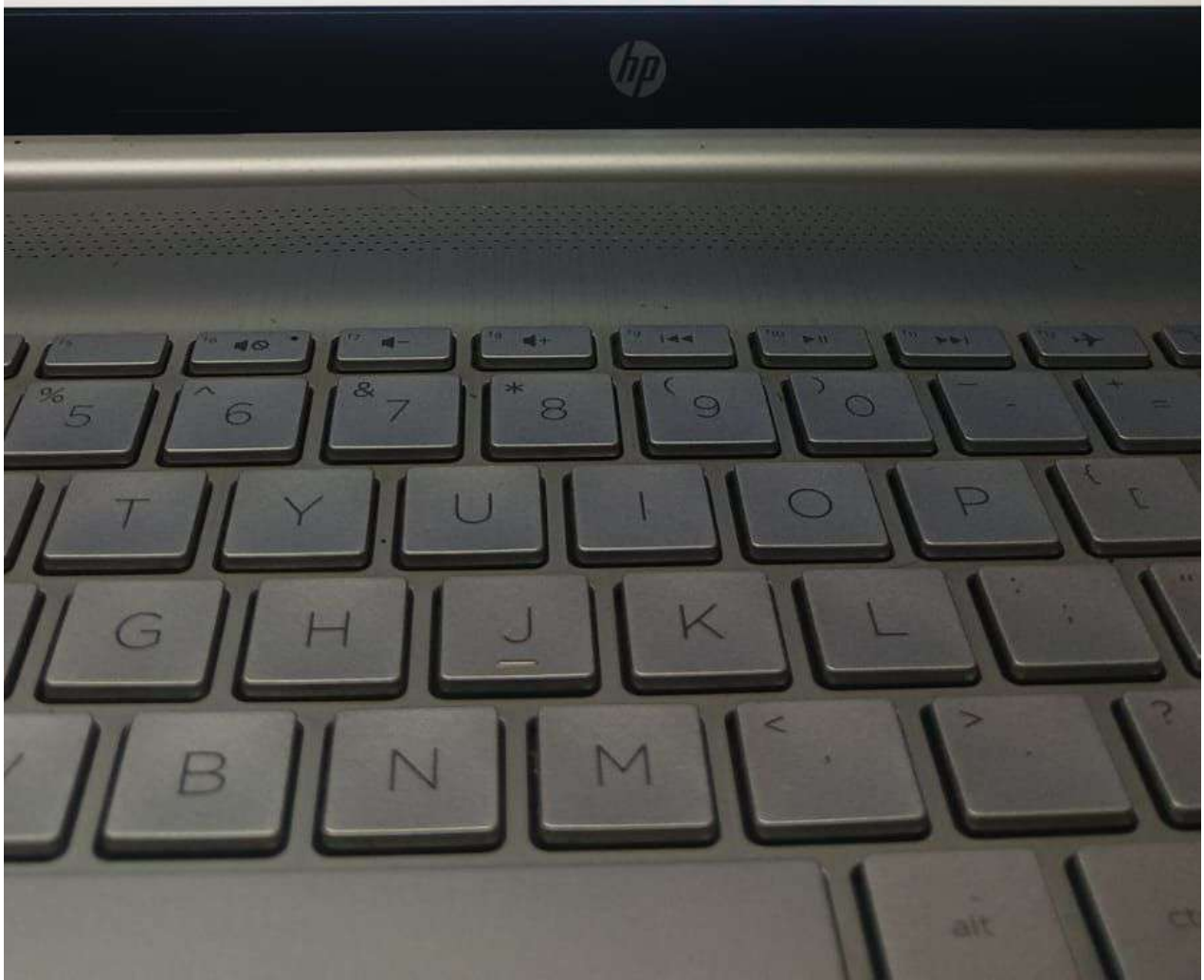
```
1 | #include <string>
2 | #include <iostream>
3 | using namespace std;
4 | int main() {
5 | string str = "hello";
6 | str[5] = '!';
7 | cout << str << endl;
8 | return 0;
9 | }
```

Answer:

- ○ The code outputs "hello!"
- ○ The code outputs "hell!"
- ○ Compiler error: Invalid string initialization
- ○ Runtime Error: Accessing an out-of-bounds array index

< Previous

Clear Answer



Concept: Briefly state the key programming concepts the question tests. If provided in the options

For the code snippet below, identify the most appropriate correction or improvement from the given options.

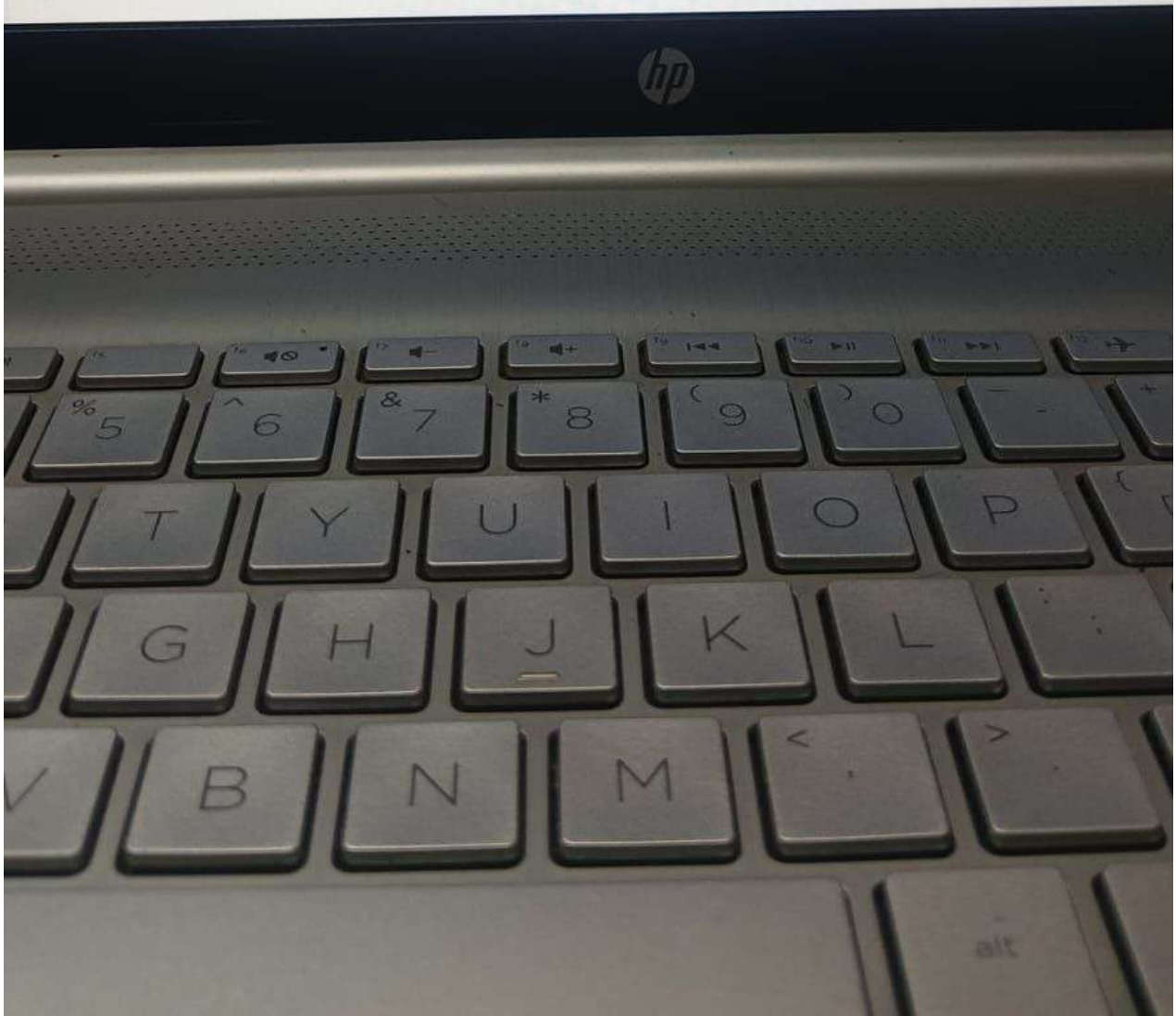
```
#include <stdio.h>
#include <stdlib.h>
int main() {
int *ptr = malloc(sizeof(int));
if (ptr == NULL) {
printf("Memory allocation failed.\n");
}
*ptr = 20;
// ... more code using ptr
return 0;
}
```

Answer:

- ○ Use realloc instead of malloc.
- ○ Cast the return value of malloc: int * ptr = (int *)malloc(sizeof(int));
- ○ The code is correct as is.
- ○ Add a call to free(ptr) at the end to release memory.

< Previous

Clear Answer



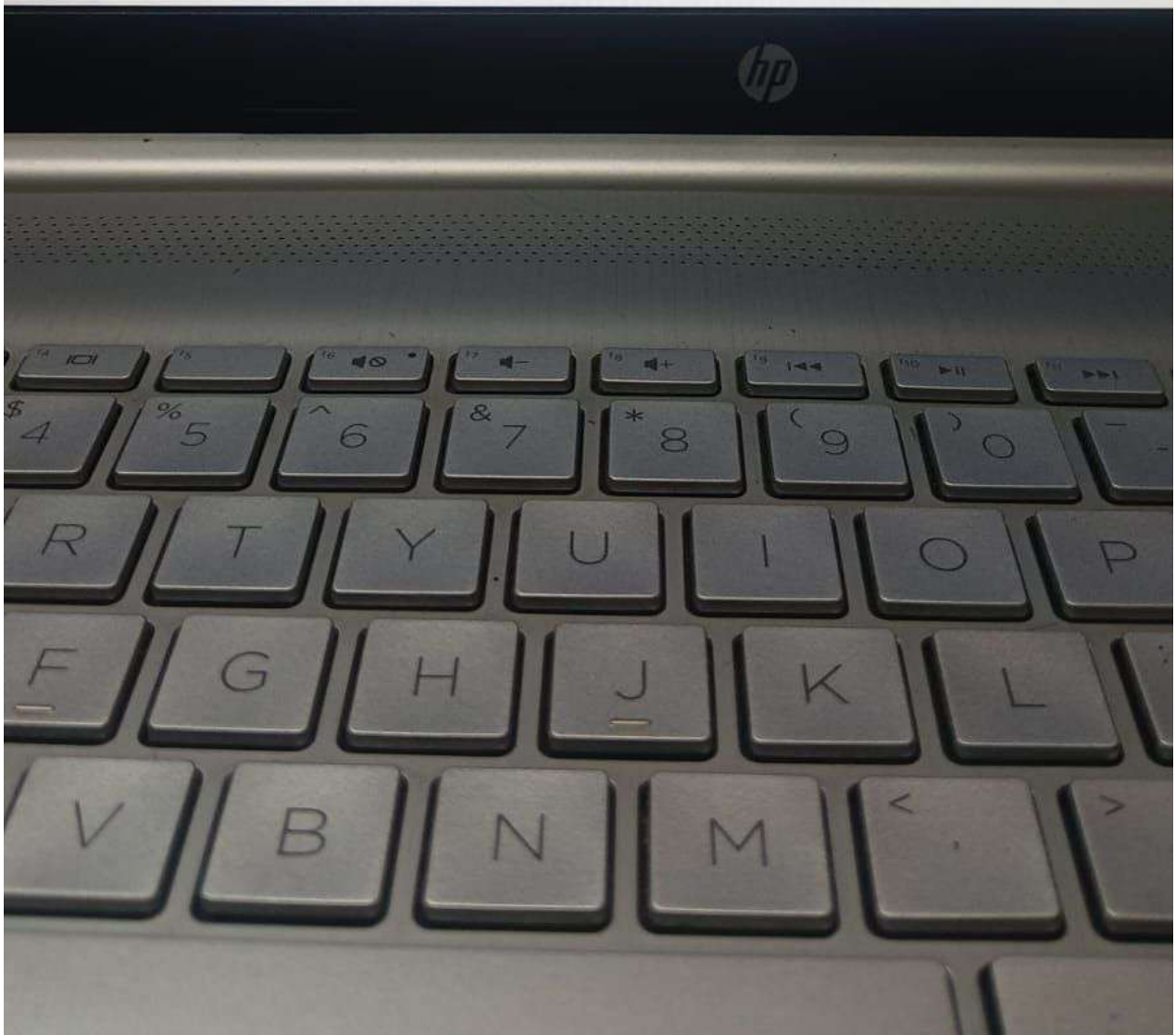
```
2 | using namespace std;
3 | void increment(int val) {
4 | val++;
5 | }
6 | int main() {
7 | int x = 10;
8 | increment(x);
9 | cout << x << endl;
10 | return 0;
11 | }
```

Answer:

- ○ Compiler error: Missing return statement in 'increment'
- ○ The code outputs 11
- ○ The code compiles, but has no output
- ○ The code outputs 10

< Previous

Clear Answer



Question: 50

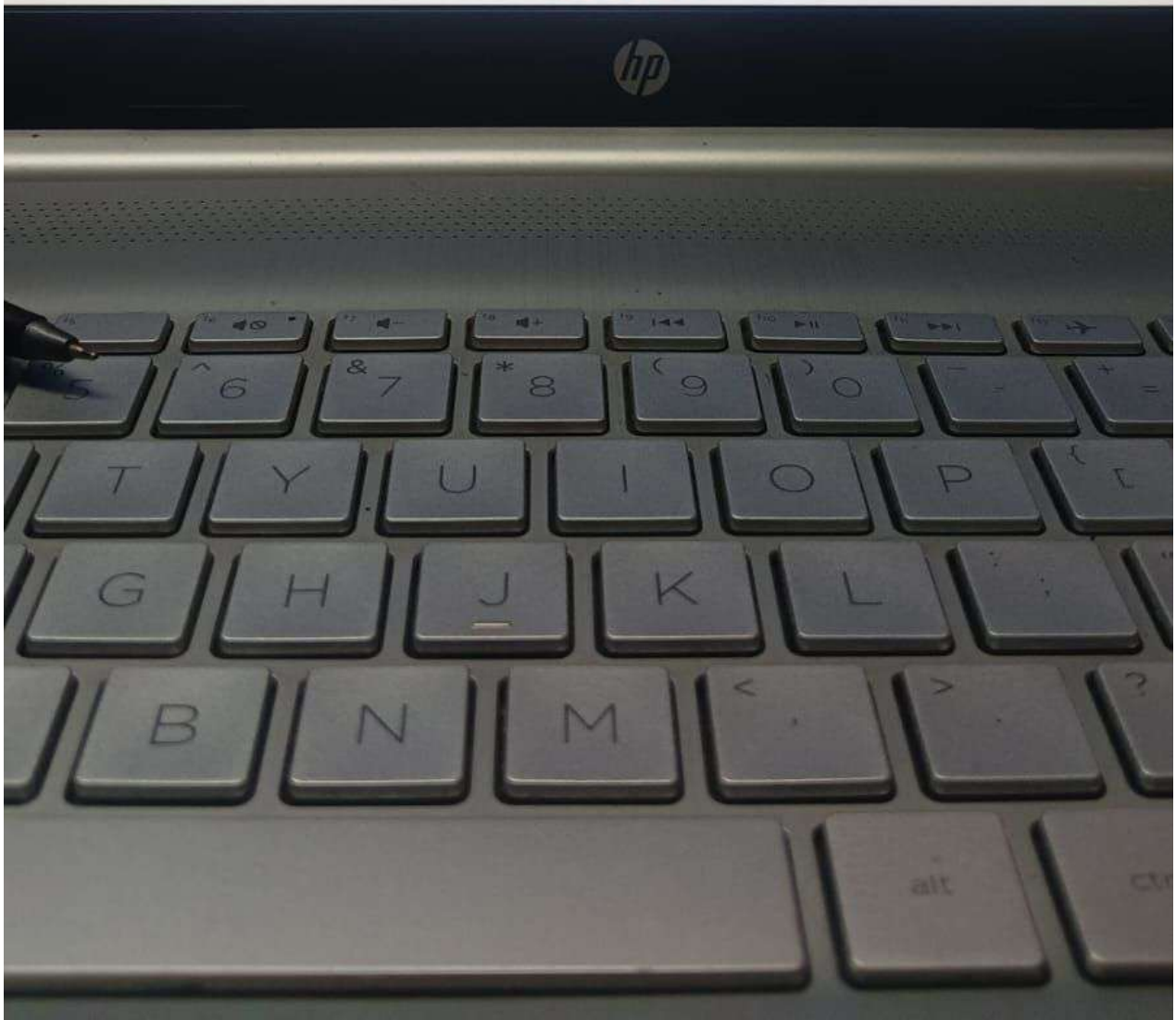
To address the "convoy effect" sometimes seen in SJF or priority-based scheduling, a potential solution is:

Answer:

- Aging – gradually increasing priority of waiting jobs.
- Strict real-time deadlines with admission control.
- Periodically reducing the priority of long processes.
- Using a lottery system for selecting the next job.

< Previous

Clear Answer



Question: 43

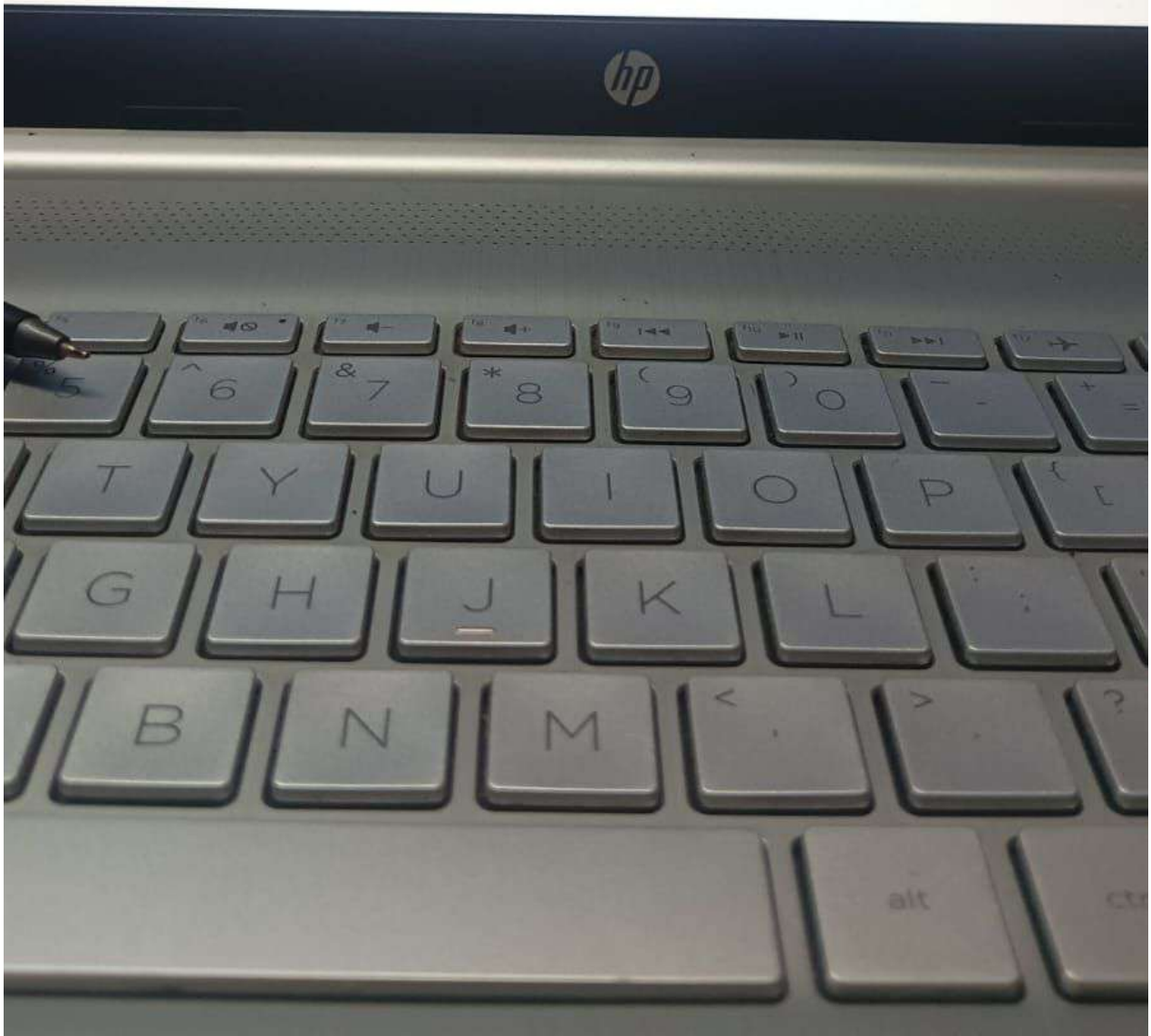
Rotational Positional Sensing (RPS) is an optimization primarily targeting:

Answer:

- ○ Minimizing power consumption of spinning disks.
- ○ Improving effective throughput over zoned-bit recording disks.
- ○ Reducing average rotational delay to read a sector.
- ○ Fault tolerance in distributed storage systems.

< Previous

Clear Answer



Review the code below and determine from amongst the provided options

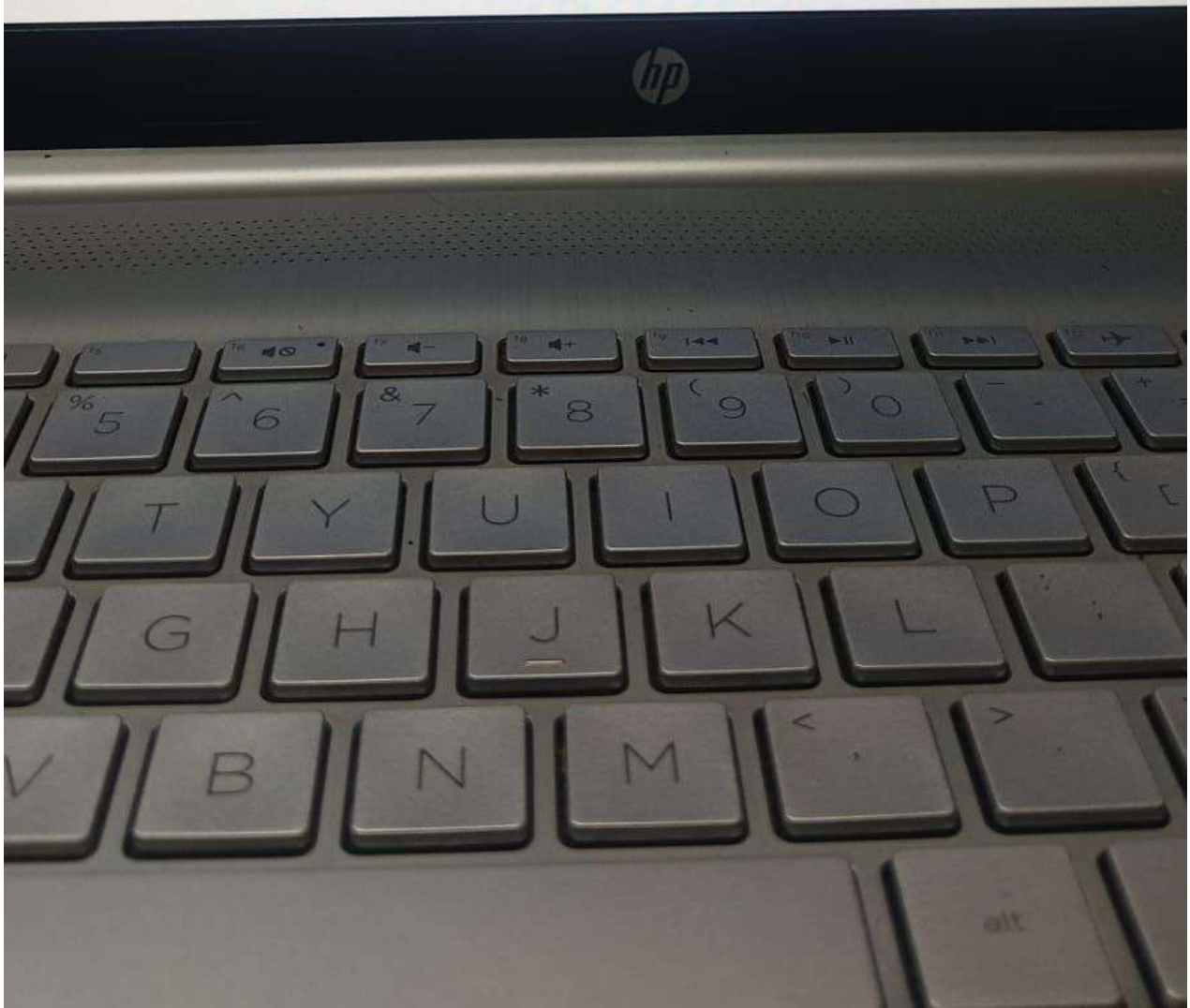
```
#include <stdio.h>
struct Point {
int x;
int y;
};
int main() {
struct Point p1 = {10, 20};
struct Point *ptr = &p1;
printf("%d\n", ptr->x);
return 0;
}
```

Answer:

- Compiler error: Invalid struct initialization
- Runtime error
- The code outputs 10
- The code outputs 20

< Previous

Clear Answer



Question: 41

Consider a cache with 64-byte cache lines, 2-way set-associativity, and a 32-bit physical address space. What's the total numbe
cache entry? (Assume 1 KB of Cache Size)

Answer:

- 26
- 6
- 21
- 19

Previous

Clear Answer

Question: 53

Which of the following best describes the concept of eventual consistency in NoSQL databases?

Answer:

- ○ Data consistency is sacrificed entirely in favor of high availability.
- ○ Data updates only become visible after a commit log is flushed to disk.
- ○ Data consistency will be achieved at some point in the future, but immediate reads may return stale data.
- ○ Data consistency is enforced immediately across all replicas.

< Previous

Clear Answer

Question: 55

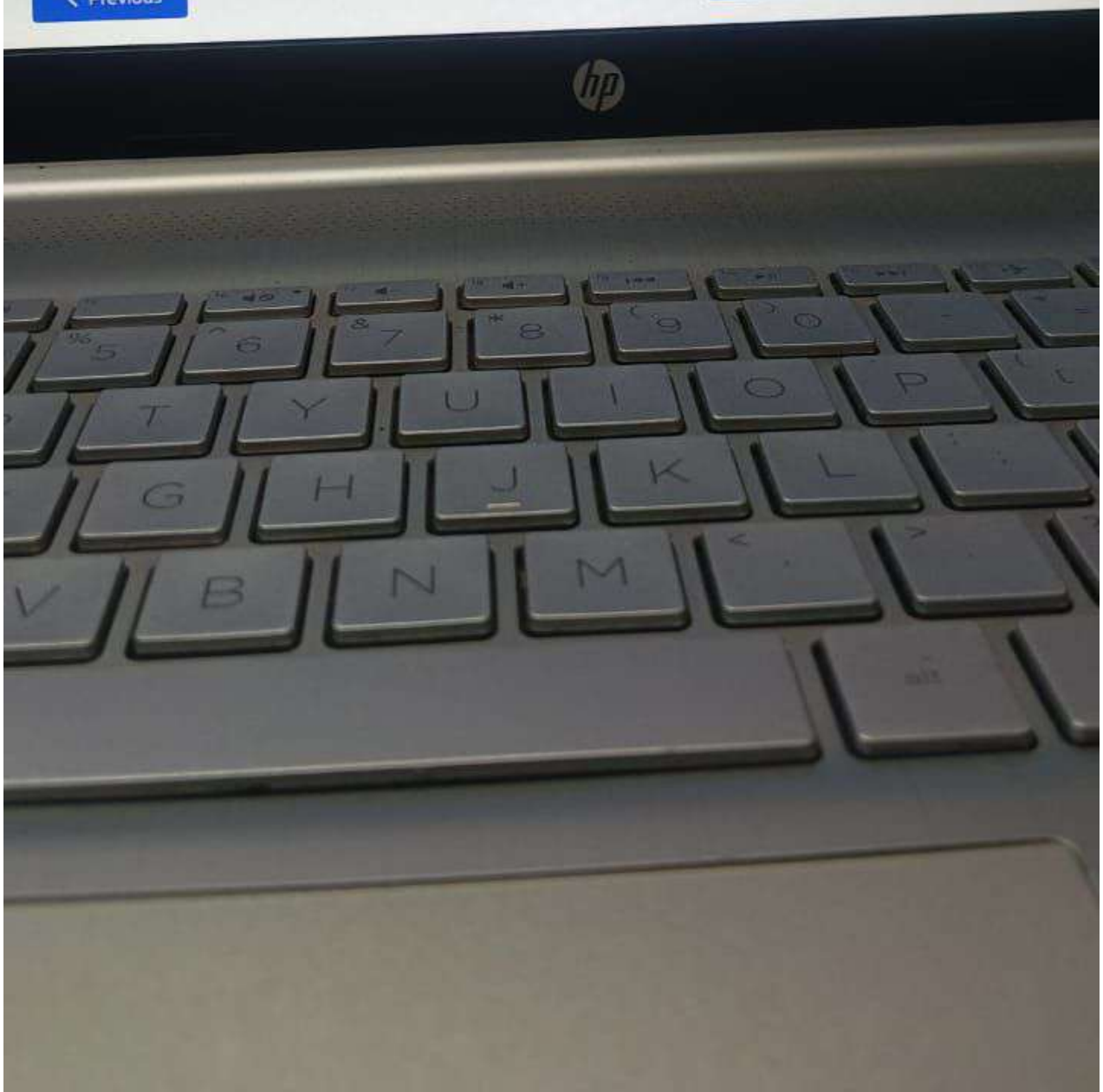
Which data model is most commonly associated with MongoDB?

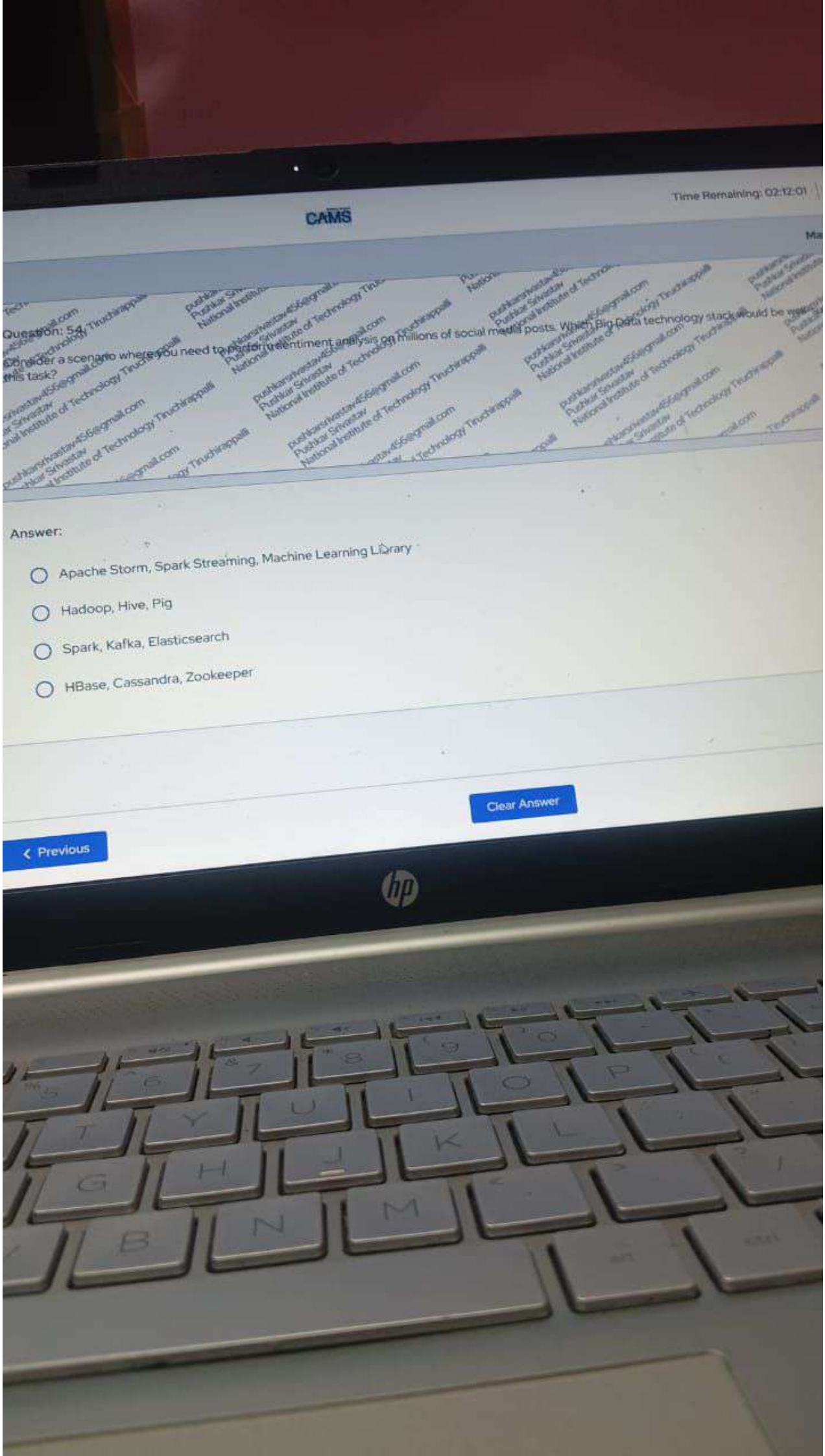
Answer:

- ○ Document
- ○ Graph
- ○ Key-Value
- ○ Column-oriented

< Previous

Clear Answer




CAMS
Time Remaining: 02:12:01
Question: 54
Consider a scenario where you need to perform sentiment analysis on millions of social media posts. Which Big Data technology stack would be most
this task?
Answer:
Apache Storm, Spark Streaming, Machine Learning Library
Hadoop, Hive, Pig
Spark, Kafka, Elasticsearch
HBase, Cassandra, Zookeeper
Clear Answer
Previous

CAMS

Time Rer

Question: 44

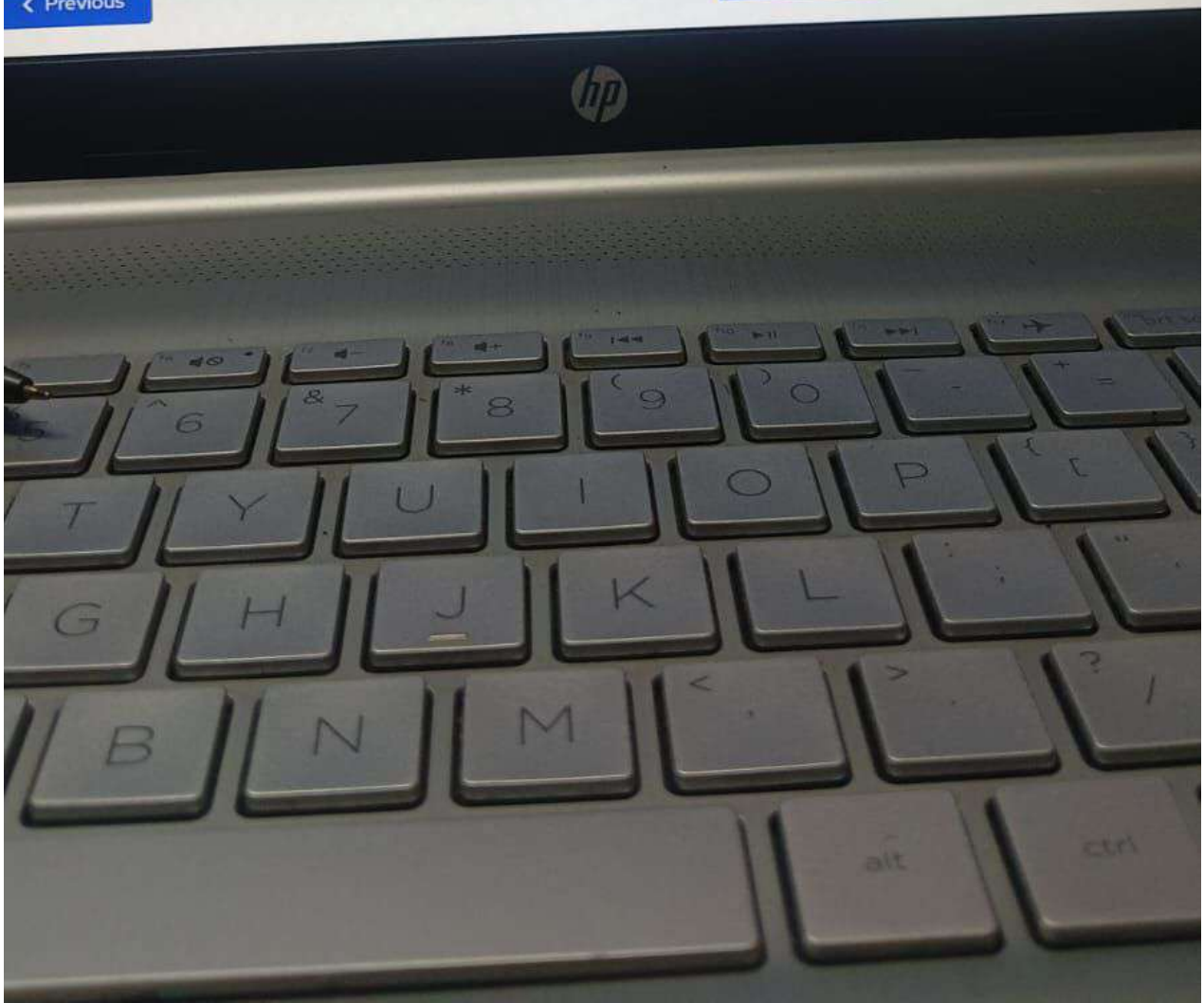
The readers-writers problem prioritizes read access when possible. Implementing a "reader priority" solution risk violating which cond

Answer:

- Bounded Waiting for writers
- Progress for readers
- No reader kept waiting unless a writer has permission.
- Mutual Exclusion for writers

Clear Answer

< Previous



Question: 42

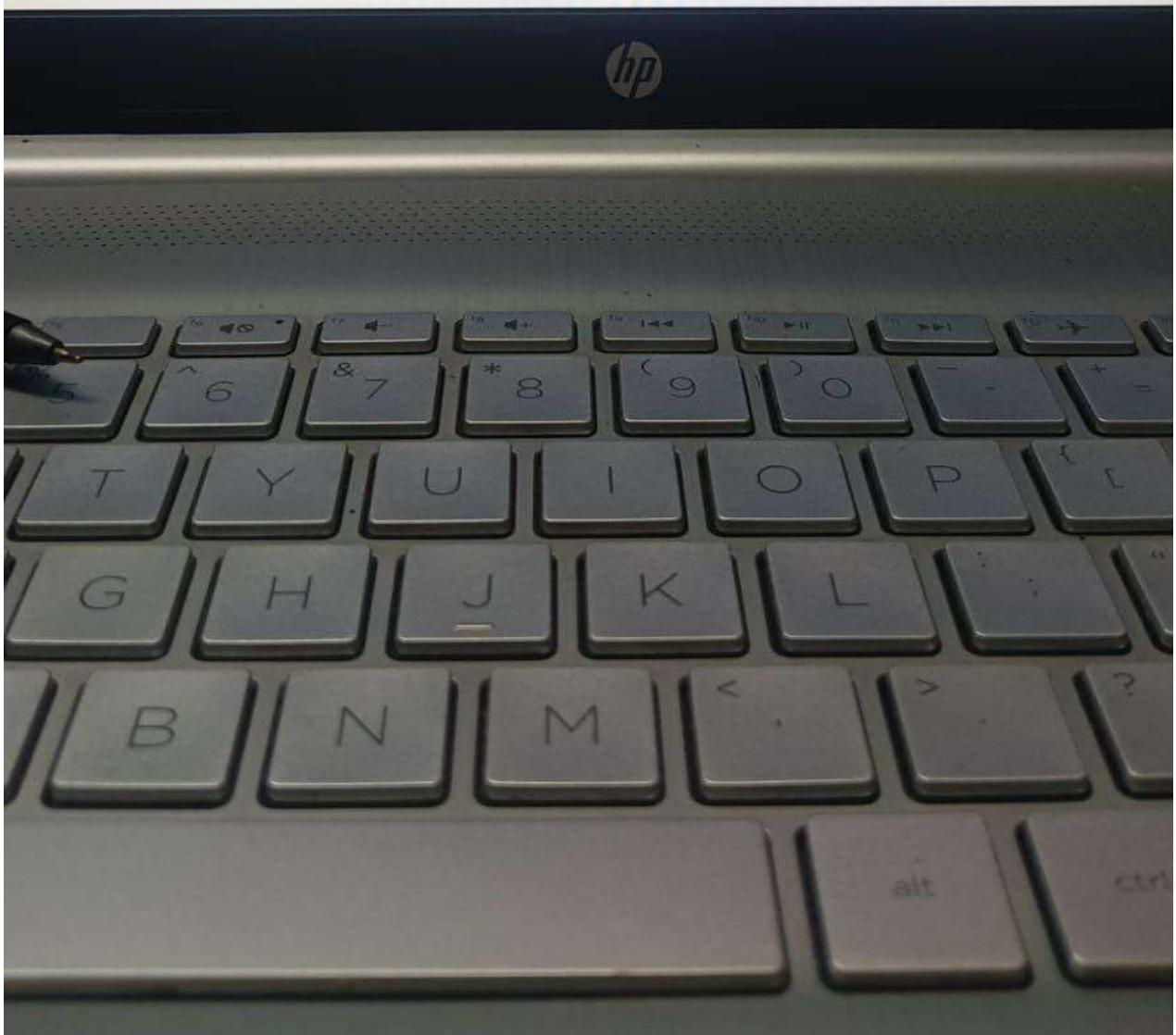
Why is interleaved memory often preferred in high-performance systems?

Answer:

- ○ It permits simultaneous access to consecutive memory locations.
- ○ It prevents memory bank conflicts, reducing bus contention.
- ○ It ensures data is distributed for memory-level parallelism.
- ○ It eliminates the need for virtual memory support.

< Previous

Clear Answer



Question: 45

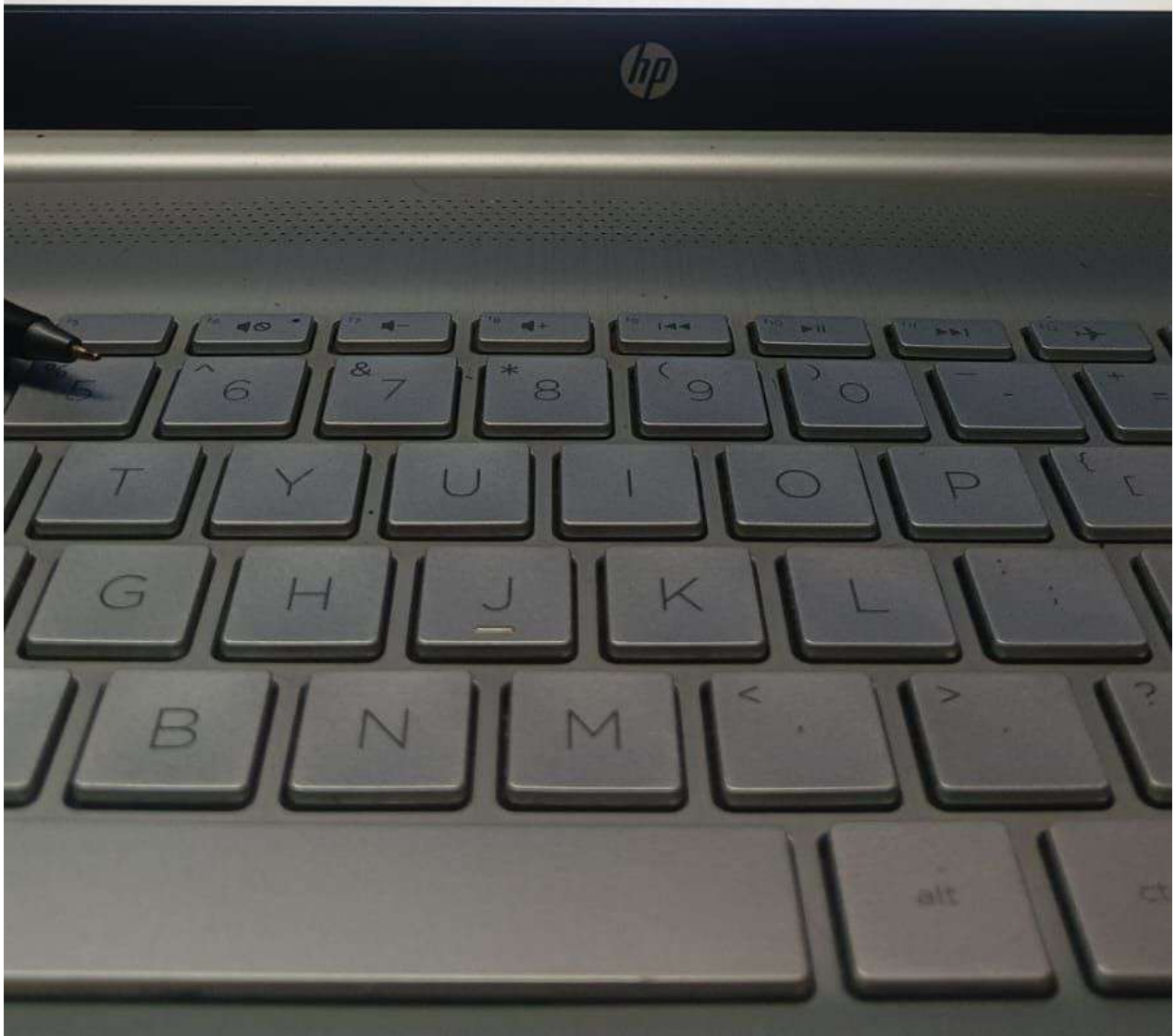
Why is interrupt-driven I/O generally favoured over programmed I/O (polling)?

Answer:

- It guarantees immediate servicing of high-priority devices.
- It eliminates the need for hardware synchronization of I/O devices.
- It requires less complex programming support from the OS.
- It allows the CPU to overlap other work while I/O occurs.

< Previous

Clear Answer



CAMS

Time Rema

Question: 46

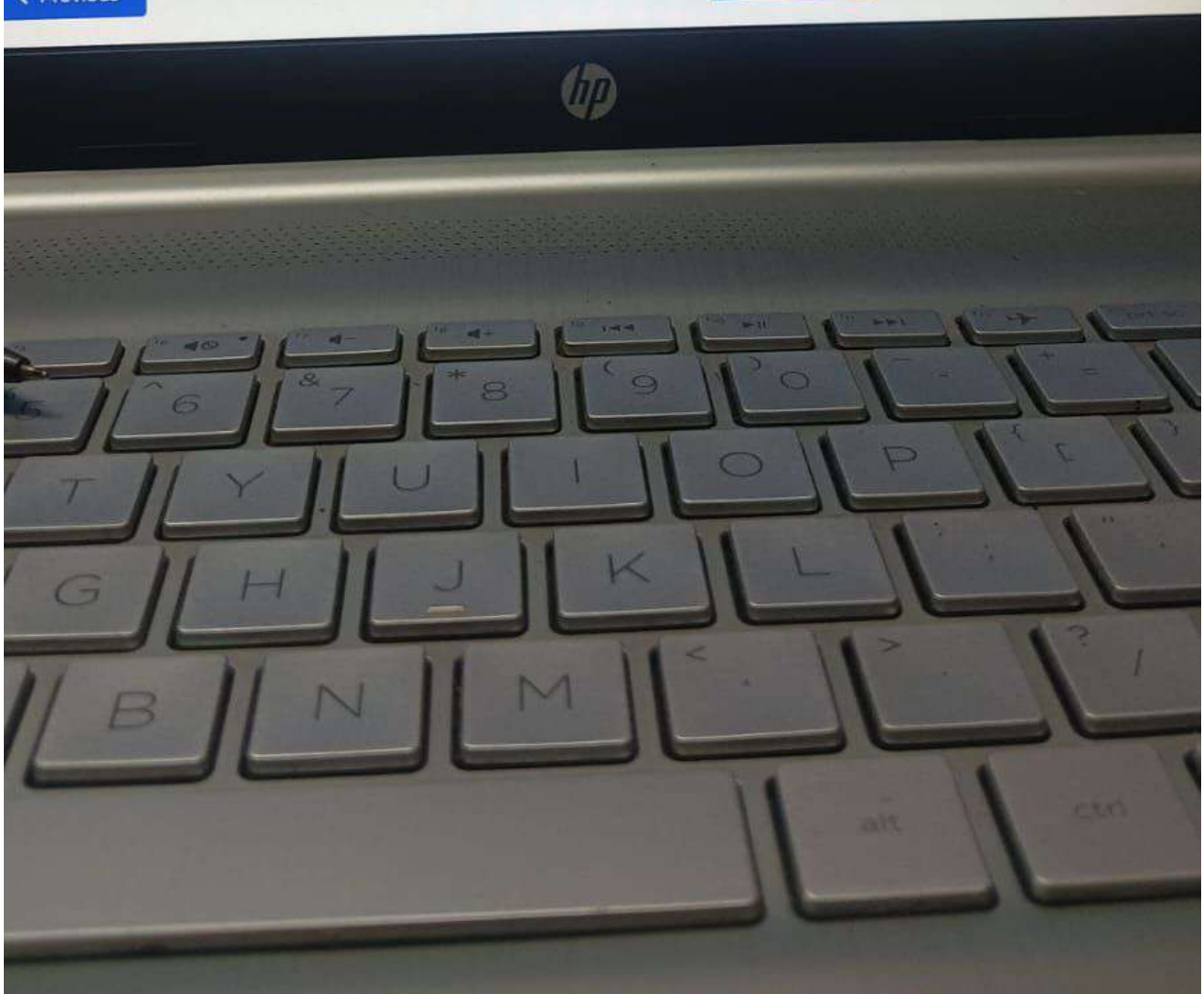
Modern CPUs often provide atomic instructions like 'test-and-set' or 'compare-and-swap'. A primary use case for these within an OS

Answer:

- ○ Context switching with reduced register saving overhead.
- ○ Handling device interrupts safely.
- ○ Implementing message passing between processes.
- ○ Building low-level spinlocks for fine-grained synchronization.

< Previous

Clear Answer



Question: 48

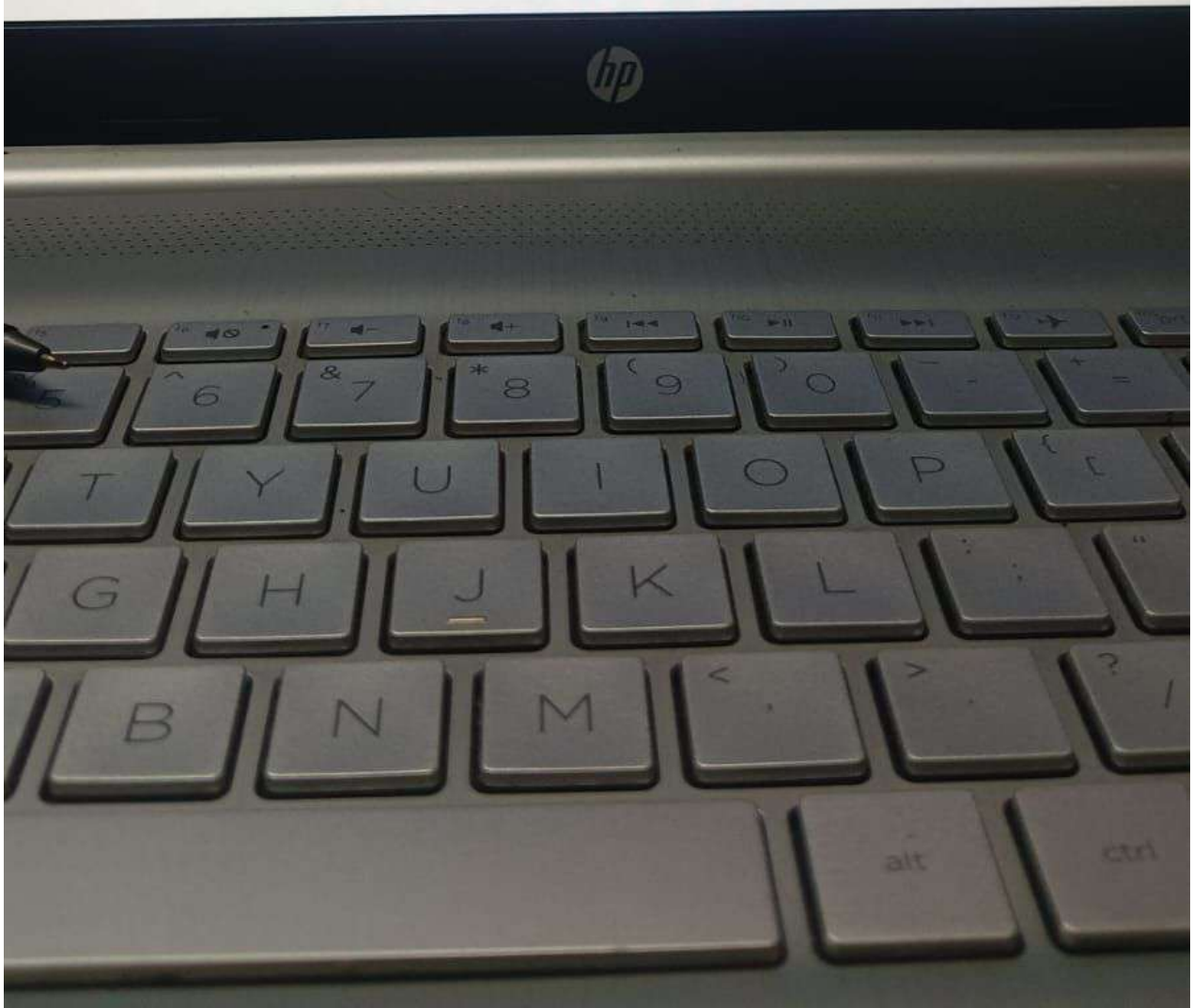
Which factor has the LEAST impact when choosing a disk scheduling algorithm for an interactive general-purpose workstatio

Answer:

- ○ Maximizing sequential read throughput for large files
- ○ Responsiveness to short file read/write requests
- ○ Minimizing average seek time
- ○ Fairness of wait times across different processes

Previous

Clear Answer



Question: 49

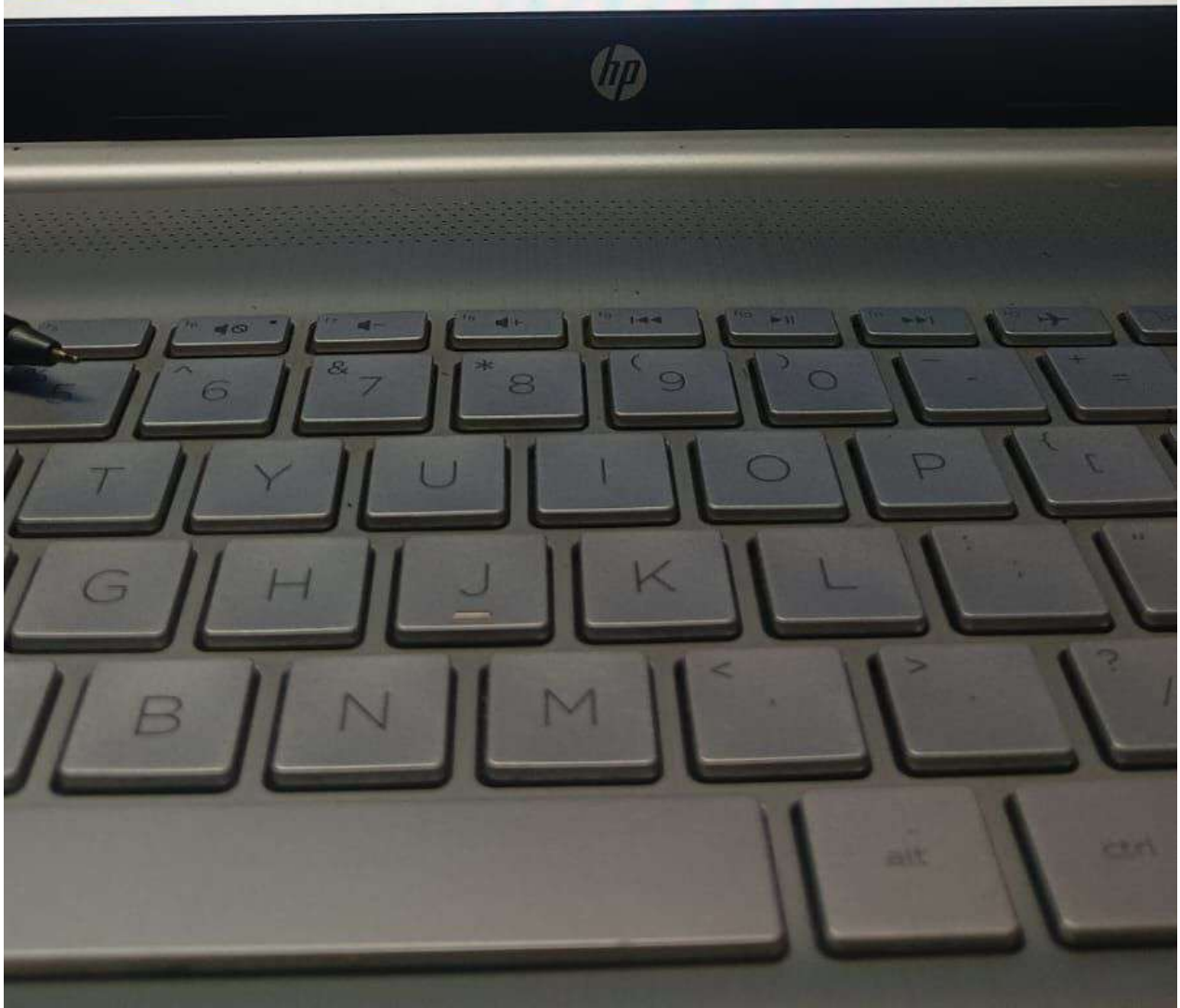
A system uses a multilevel feedback queue, but processes seem stuck at lower priority levels. A likely cause is:

Answer:

- ○ I/O operations aren't releasing the CPU as expected.
- ○ The time quantum at the upper priority level is too short.
- ○ The time quantum at the upper priority level is too long.
- ○ There are not enough CPU resources for the workload.

< Previous

Clear Answer



CAMS

Time Remai

Question: 82

You need an algorithm to determine if there exists a path between two vertices in a directed graph. Which is the most suitable choice?

Answer:

- ○ Prim's algorithm
- ○ Any graph traversal algorithm (DFS or BFS)
- ○ Dijkstra's Algorithm
- ○ Kruskal's algorithm

Previous

Clear Answer

CAMS

Time Remaini

Question: 36

You need a self-balancing binary search tree. Which guarantees worst-case O(log n) operations at the expense of potentially complex rot

Answer:

- Red-black tree
- AVL Tree
- Splay tree
- B-tree

Clear Answer

Previous

Question: 47

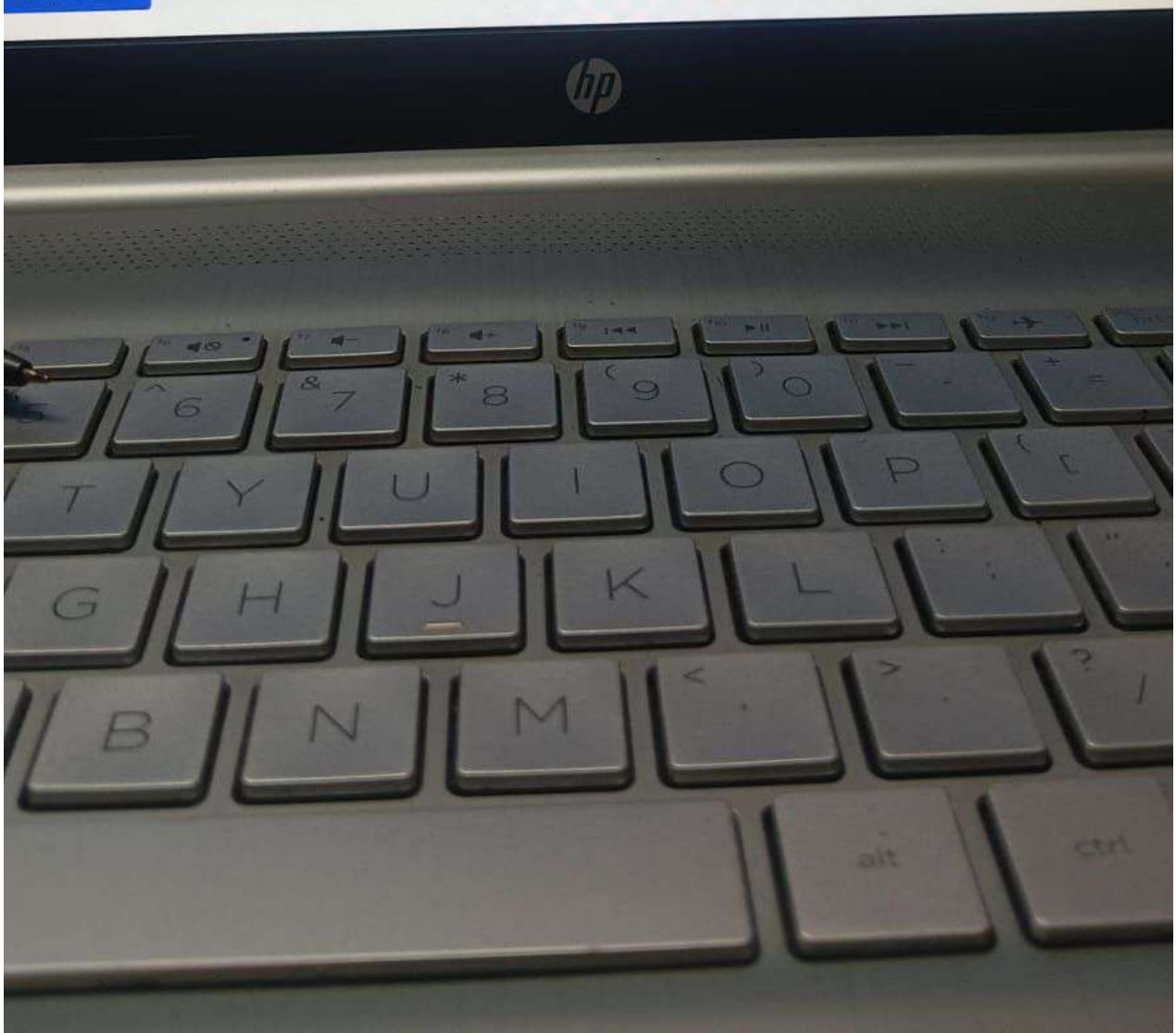
In a paged system, the Page Table Base Register (PTBR) is crucial. When a process switch occurs, which action MUST be taken?

Answer:

- ○ All of the mentioned
- ○ The CPU must enter privileged mode to access the PTBR.
- ○ All entries in the TLB must be invalidated.
- ○ The PTBR must be updated to point to the new process's page table.

< Previous

Clear Answer



Question: 32

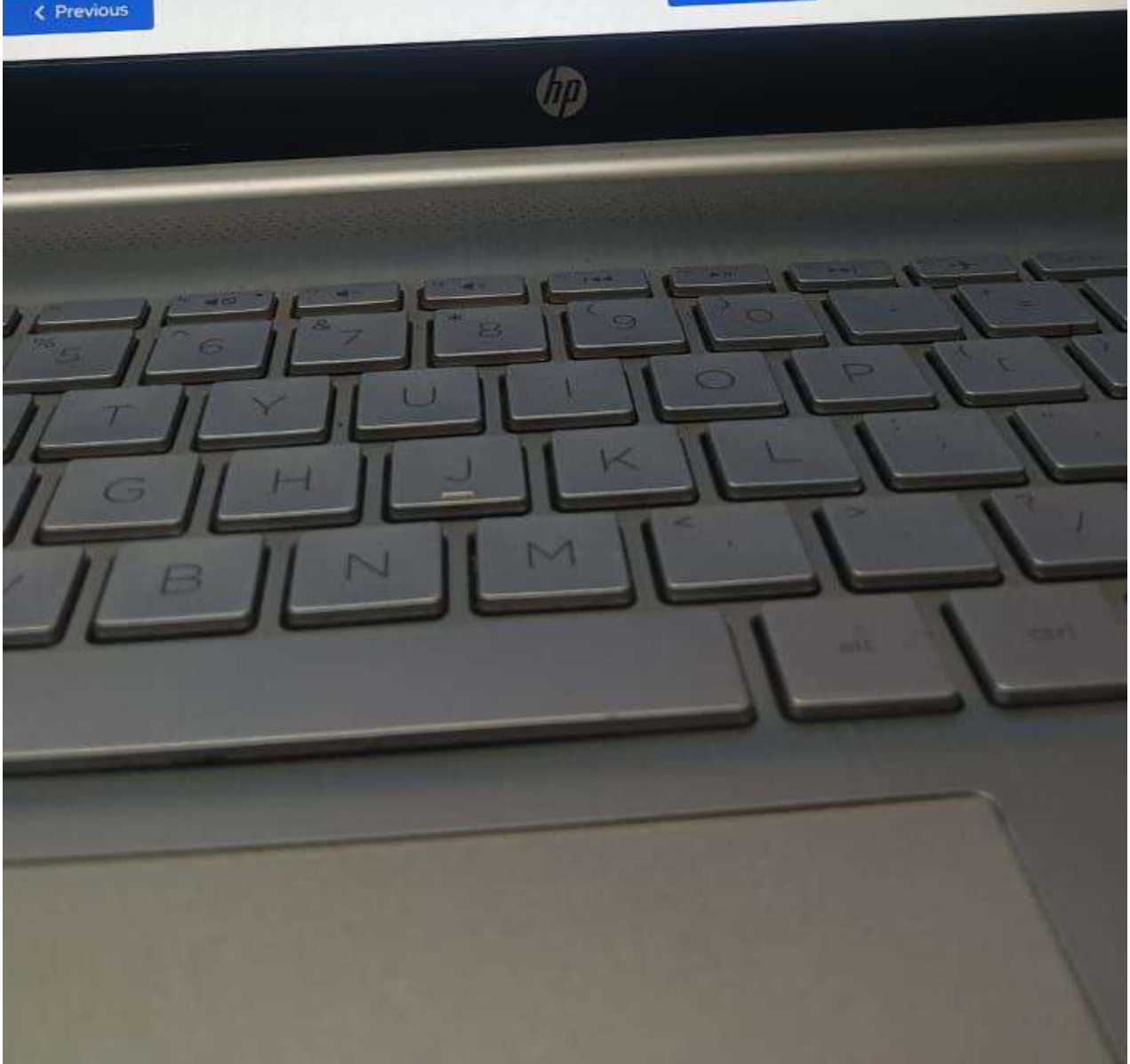
For pattern matching where the pattern may contain wildcards or special characters, which algorithm outperforms simple brute-force?

Answer:

- ○ Knuth-Morris-Pratt (KMP) algorithm
- ○ Naive string search
- ○ Aho-Corasick algorithm
- ○ Rabin-Karp algorithm

Clear Answer

< Previous



Question: 35

What data structure underlies the implementation of most version control systems like Git for tracking changes to a file over time?

Answer:

- ○ Directed acyclic graph (DAG)
- ○ Bloom filter
- ○ Merkle tree
- ○ Linked list

Clear Answer

‹ Previous

Question: 34

Which algorithm would you use to generate all possible permutations of a given array?

Answer:

- ○ Depth-first search (with backtracking)
- ○ Merge sort
- ○ Heap's algorithm
- ○ Topological sort

Previous

Clear Answer

DBMS

a) integrity and analytics

b) Time variant attributes

c) Data consistency is compromised

54) — Apache Storm, Spark streaming

55) Docum

56) Transform

57) Batch process of data, unstru

58) consistency, Availa, Partition Tole

59)

60) Fixed schema and the vec
horizontal scale
fixed schema

Question: 33

You need a data structure for a real-time leader board with these operations:

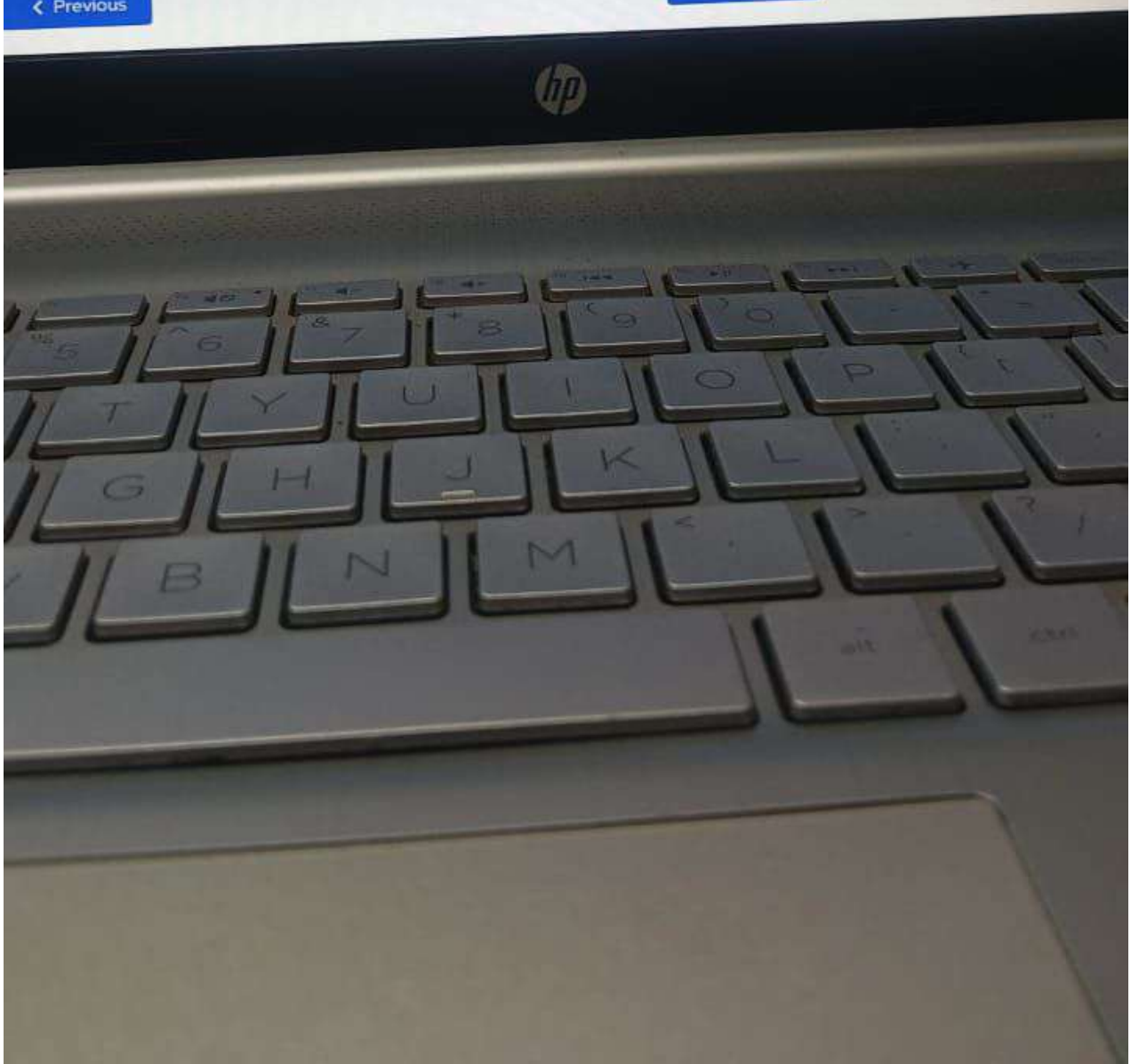
insertScore(player, score), deleteScore(player), and getTopK(k). Which is the MOST efficient combination?

Answer:

- ○ Skip list for insertion/deletion, min-heap for getTopK.
- ○ Self-balancing BST (e.g., AVL tree) for insertion, hash table for deletion.
- ○ Max-heap for insertion/lookup, array for deletion.
- ○ Sorted array for insertion, hash table for lookup.

Clear Answer

< Previous



Question: 52

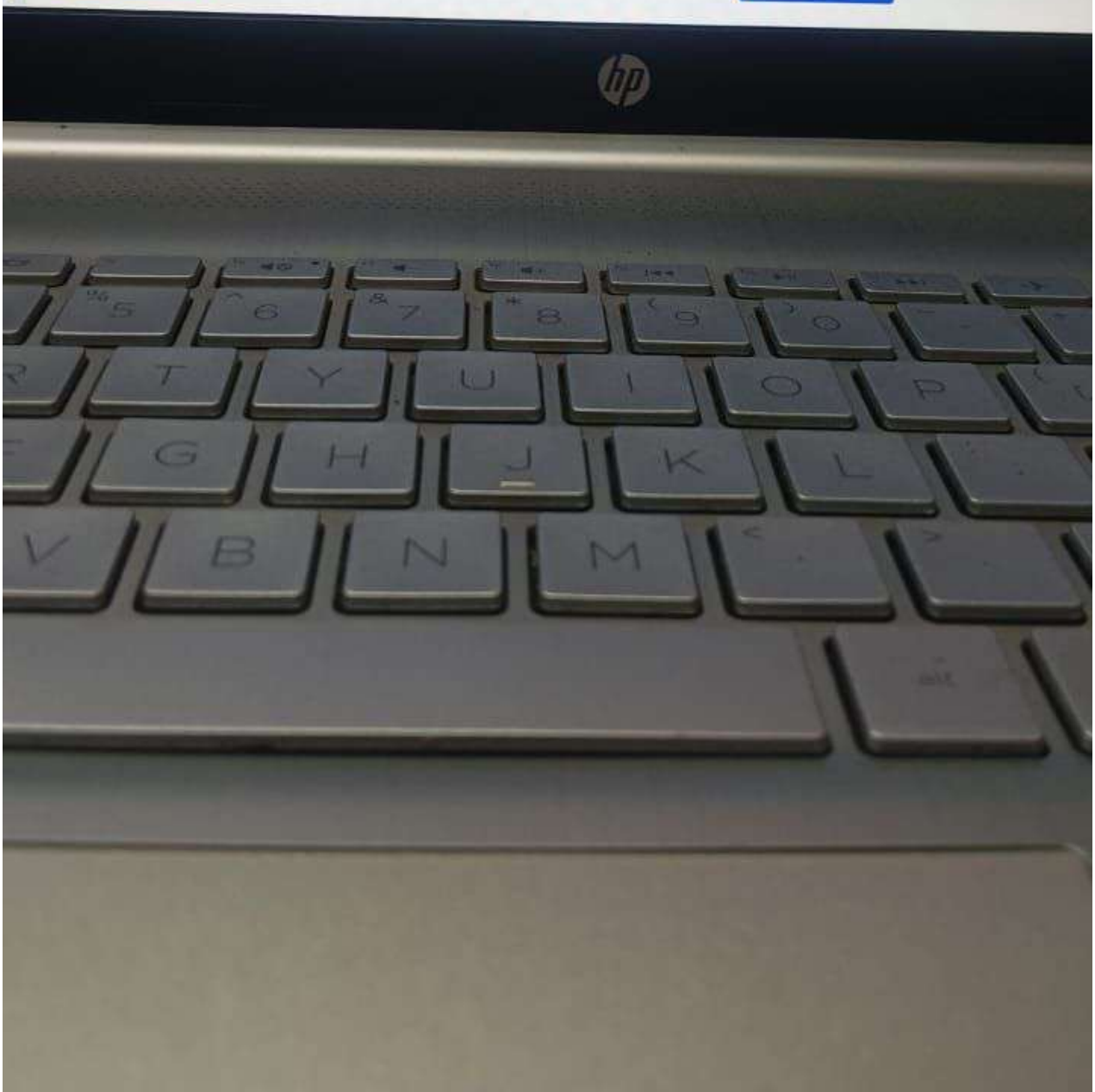
Fact tables in a data warehouse often contain:

Answer:

- Quantitative measures and metrics
- Time-variant attributes that describe entities
- Primary and foreign keys
- Textual descriptions of entities

Previous

Clear Answer



Question: 40

What kind of tree allows constant-time access to the minimum AND maximum element within it?

Answer:

- Binary heap
- Binary search tree (unmodified)
- AVL tree
- Cartesian Tree

Clear Answer

Previous

CAMS

Time Remaining: 02:04:18

Question: 38

There exists an algorithm with time complexity O(n!) to solve a problem. Processing an input of size n=5 takes 0.2 seconds. Estimate the MAXIMUM you could reasonably process in under 1 minute.

Answer:

- n = 8
- n = 9
- n = 7
- n = 6

Clear Answer

‹ Previous

simulated annealing
Dynamic prog
Adjacency matrix
Trie
algorithm (merge sort)

26) O(m√n)
27) Encapsulation
28) count min sketch
29) Rabin karp algorithm
30) polymorphism
31) serialization
32) Aho-corasick algo
33) ---
34) Depth first search
35)
36) DAG
37) AVL tree
38) DFS and bfs
39) n = 6
40) CSR format
Binary heap

CAMS

Time Remain

Question: 31

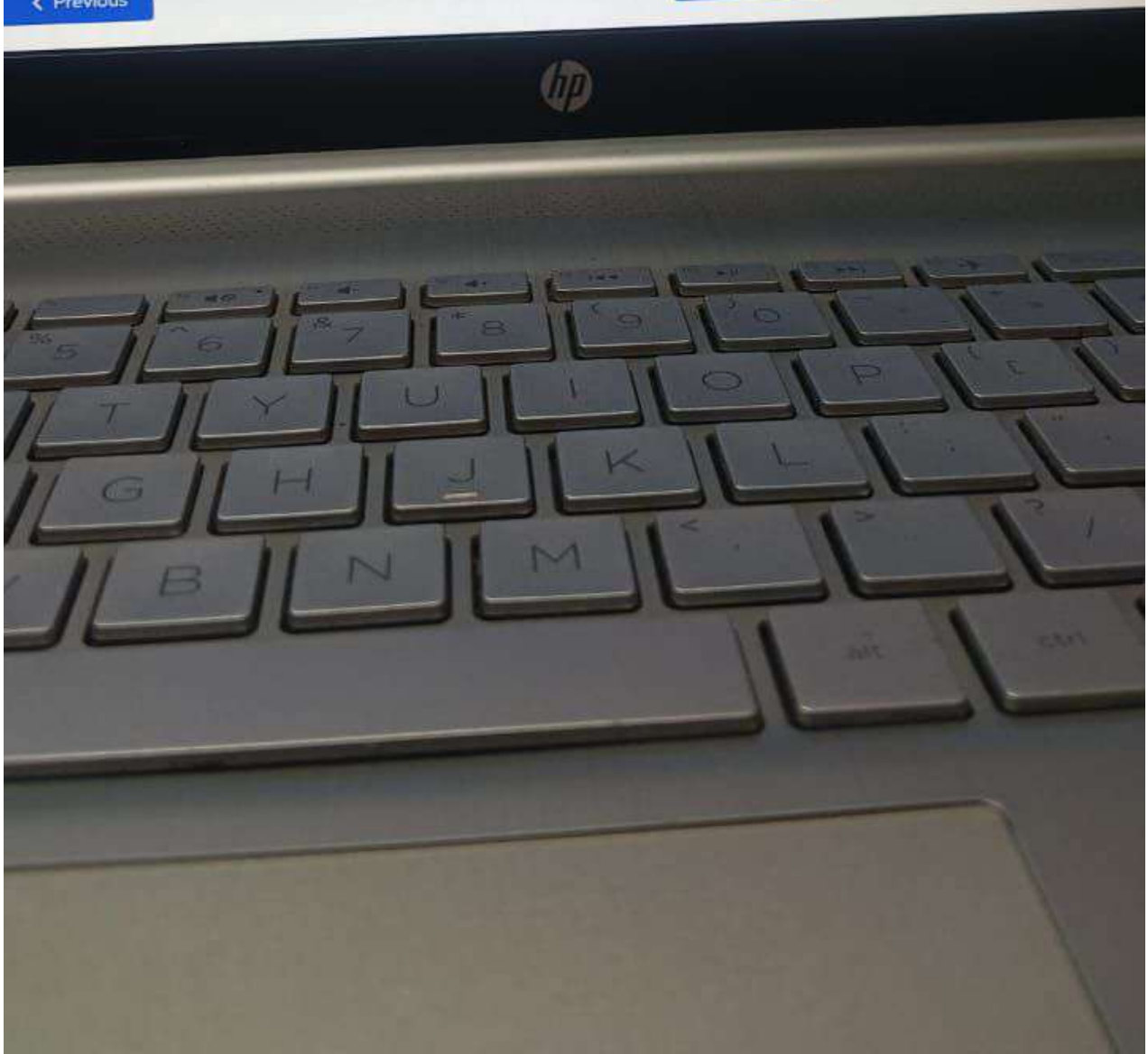
An algorithm demonstrates logarithmic growth. If processing 10 items takes 3 milliseconds, approximately how long would 1000 items tak

Answer:

- 300 milliseconds
- 100 milliseconds
- 9 milliseconds
- 30 milliseconds

Previous

Clear Answer



CAMS

Time Remain

Question: 30

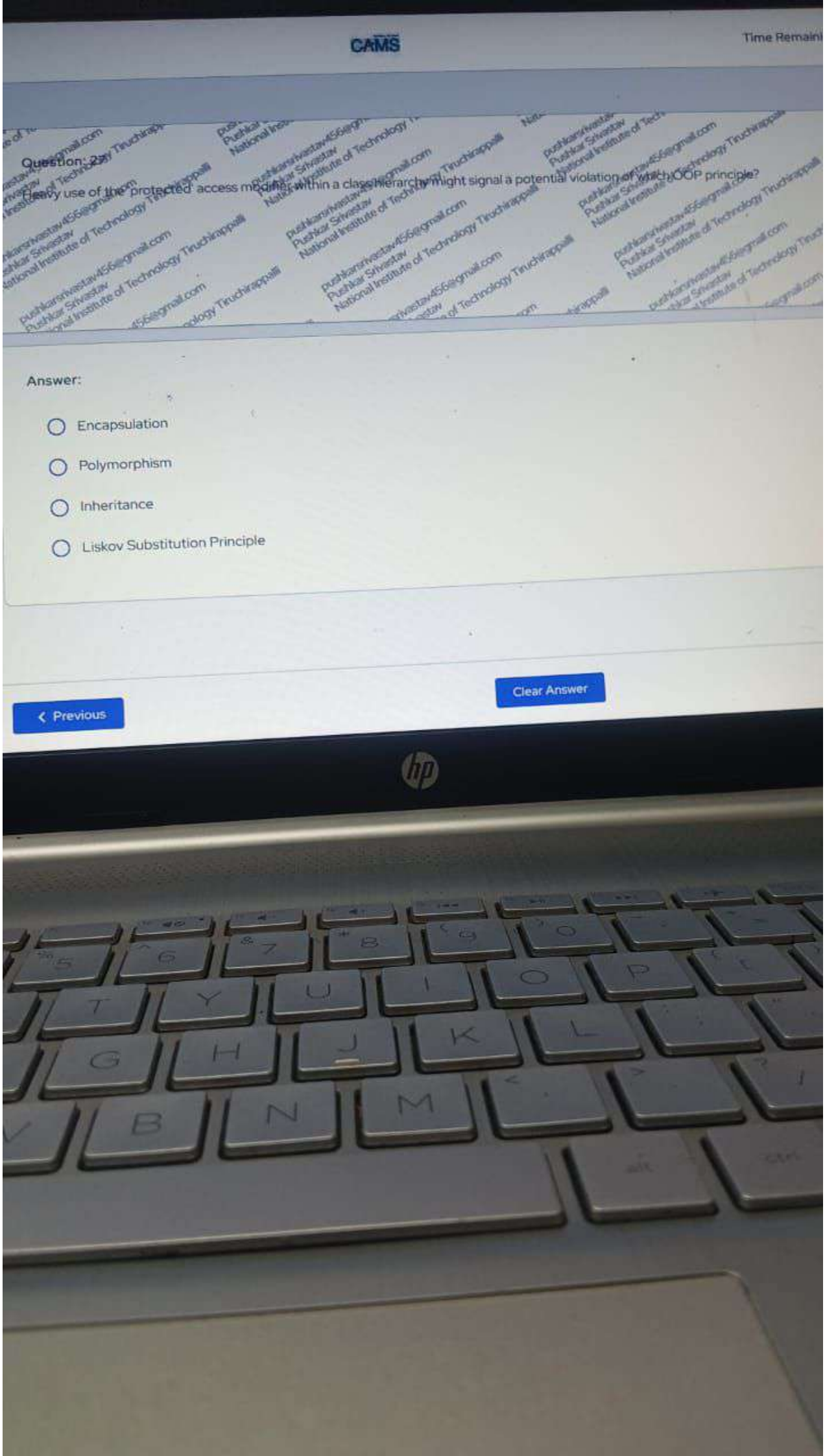
Which principle is BEST demonstrated by providing multiple implementations of a 'sort()' method, tailored to different datatypes within

Answer:

- Polymorphism
- Abstraction
- Inheritance
- Encapsulation

Previous

Clear Answer


CAMS
Time Remaini
Question: 27
Heavy use of the 'protected' access modifier within a class hierarchy might signal a potential violation of which OOP principle?
Answer:
Encapsulation
Polymorphism
Inheritance
Liskov Substitution Principle
Previous
Clear Answer

Question: 29

To search for an extremely rare pattern in a massive text corpus, with failure being common but early rejection of non-matches desirable, which algorithm excels?

Answer:

- Knuth-Morris-Pratt Algorithm
- Rabin-Karp Algorithm
- Boyer-Moore Algorithm
- Naive string search

Previous

Clear Answer

Question: 28

To track the real-time frequency of words in a streaming text feed, which data structure keeps memory usage reasonable while allowing

Answer:

- Suffix array
- Min-heap
- Self-balancing binary search tree
- Count-min sketch

Clear Answer

Previous

CAMS

Time Remaining:

Question: 39

Implement a sparse matrix efficiently (many zero elements). Which is most memory-conscious but preserves reasonable element access?

Answer:

- ◯ Compressed sparse row (CSR) format
- ◯ Hash table mapping coordinates to values
- ◯ List of key-value pairs (coordinates and value)
- ◯ Store in a standard 2D array

Clear Answer

‹ Previous

CAMS

Time Remain

Question 26

You have a recursive algorithm with the recurrence relation T(n) = 2T(n/4) + n. Using the Master Theorem, approximate its time comple

Answer:

- O(n log n)
- O(n $\log^2$ n)
- O(n^2)
- O(n $\sqrt{n}$)

Clear Answer

Previous

CAMS

Time Rem

Question: 25

Two sorting algorithms are considered:

Algorithm A: Best case O(n), Average/Worst case $O(n^2)$

Algorithm B: All cases O(n log n)

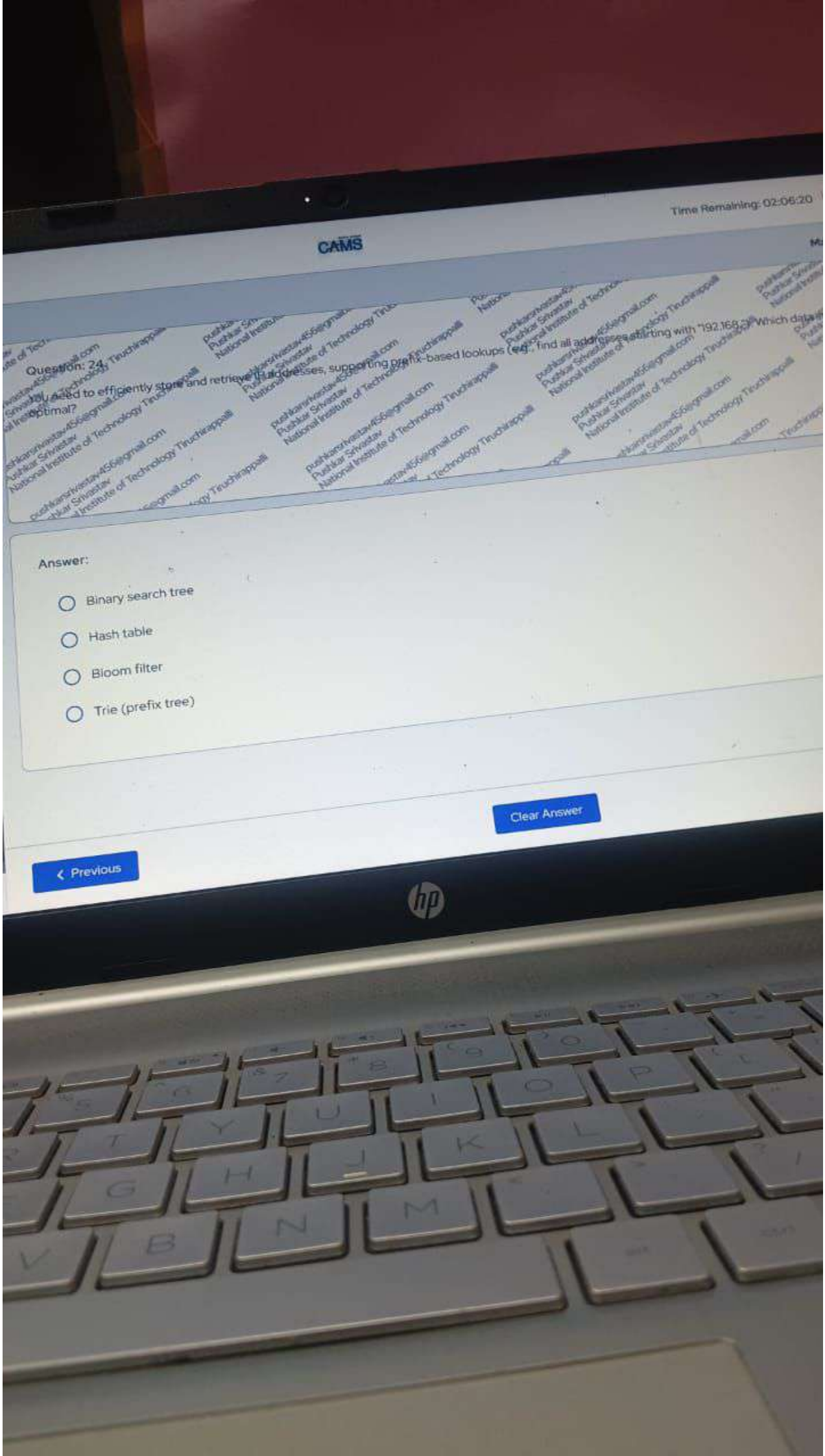
When sorting extremely large, almost pre-sorted datasets, which would likely be faster?

Answer:

- ○ Impossible to know without testing on the specific data
- ○ Algorithm B (e.g., merge sort's consistency wins)
- ○ They would perform roughly the same
- ○ Algorithm A (e.g., insertion sort would excel)

Clear Answer

< Previous


CAMS
Time Remaining: 02:06:20
Question: 24
optimal?
Answer:
Binary search tree
Hash table
Bloom filter
Trie (prefix tree)
Clear Answer
Previous

Question: 21

To approximate the solution to the notoriously difficult Traveling Salesman Problem, which heuristic approach often yields decent re

Answer:

- Simulated annealing
- 2-opt or 3-opt local search
- Branch and bound
- Backtracking

Clear Answer

Previous

Question: 23

Which graph representation is generally best for a dense graph (where most potential edges exist)?

Answer:

- Incidence matrix
- Adjacency matrix
- Edge list
- Adjacency list

< Previous

Clear Answer

CAMS

Time Remaining: 02

Question: 22

A computationally expensive optimization problem seems intractable. To compromise, you consider an approach that rapidly produces good b
necessarily optimal solutions. This describes

Answer:

- Divide and Conquer
- Dynamic Programming
- Approximation Algorithm
- Greedy Algorithm

Clear Answer

Previous