**Quiz Answers**

Paper 1: Code Confusion

1. B) 12

2. A) 28

3. A) True

4. A) 0 1 2 3 4

5. D) None of the above

Paper 1: Internet Lingo

6. A) This code is exceptionally well-written and effective, and I'm being sincere.

7. A) The pull request is mediocre/average

8. A) Looks Good To Me

9. A) Take a break from the computer and go outside

10. A) Your code is authentic, solid, and impressive

Paper 1: College Inside Jokes

11. A) Strong negative correlation: cleanliness decreases as finals approach

12. A) Approaches zero as the deadline approaches

13. A) No amount can substitute - the answer tends to infinity

14. A) Rubber duck debugging

15. C) 8 - A normal part of the CS experience

Paper 1: General Nonsense

16. C) Semicolons spontaneously transform into Greek question marks

17. A) Your soul is slowly drained with each character typed

18. C) It throws an UnobservedException Exception

19. D) No one - it's actually a sign something is terribly wrong

20. B) You create a quantum singularity that solves P=NP

21. B) Performing the sacred Office Space ceremony with a baseball bat

22. B) Directly proportional to the urgency of the deadline

23. C) Two — one to do it and one to explain why the previous approach was better

24. A) It's a legendary key that only appears to truly desperate users

25. D) Temporarily yes, permanently no

**Quiz - Paper 2: Answer Key**

1. **A) malloc() allocates uninitialized memory while calloc() initializes memory to zero**
2. **D) Both A and C**
3. **C) Causes undefined behavior**
4. **A) 10**
5. **D) Both A and B**
6. **B) "I'll still let you stare at this code for 20 minutes before you notice the semicolon is missing"**
7. **C) "The answer is on page 573 of a 1200-page PDF I haven't opened since 2007"**
8. **A) "When the heat death of the universe makes this bug irrelevant"**
9. **A) Time itself warps to create a reality distortion field where "quick" means "three more sprints"**
10. **B) "It's 0.02% faster but completely unmaintainable"**
11. **D) "All of the above, and I'm not even sorry about it"**
12. **C) Planck time units — it essentially disappears instantly**
13. **D) Useful comments you didn't bother to write**
14. **D) Single letters that give absolutely no hint about what they point to**
15. **A) Adding 500 printf statements and slowly removing them one by one**
16. **B) False, but it helps you remember what your code does five minutes after writing it**
17. **A) Mythological artifact**
18. **A) Yes, that's literally what waking up disoriented is**
19. **C) In very specific error handling cases that you'll have to defend in every code review**
20. **C) Schrödinger's buffer: it's both overflowed and secure until measured**
21. **A) Because pointers can sense fear even without visual confirmation**
22. **D) Repeatedly muttering "it's just a segfault, it's just a segfault" to build emotional resilience**
23. **B) Decreases it, unless your team consists entirely of Star Trek convention attendees**
24. **A) Off-by-one errors caused by the universal confusion about whether arrays start at 0 or 1**
25. **C) It's inversely proportional to the programmer's understanding of each header**

**Quiz - Paper 3: Answer Key**

1. **C)** None of the above
2. **B) Buffer overflow: allocated 10 bytes but writing more**
3. **C) Compilation error**
4. **A) gets()**
5. **A) 3 5 2**
6. **A) My code is crashing before submission, the professor will certainly fail me for real**
7. **A) You've reached the final, most desperate debugging phase**
8. **A) Your code works but looks like it was written by a sleep-deprived panda**
9. **D) All of the above**
10. **D) All of the above, in that exact order**
11. **A) False - printf is the sacred debugging tool of every VTU student**
12. **B) Each 1% below 85% attendance requires 1 additional hour of cramming**
13. **C) Dennis Ritchie's Blessing**
14. **A) Directly proportional to how close you are to finishing**
15. **C) Exponentially increasing as the submission deadline approaches**
16. **A) "I'm using Turbo C++ and you're using GCC"**
17. **C) Quickly add 50 more printf statements**
18. **C) No comments and barely working code**
19. **B) "I made it 1% faster and 200% harder to understand"**
20. **D) All of the above**
21. **D) All of the above, in increasingly desperate succession**
22. **A) Run it and see if your computer crashes**
23. **B) Memorize the expected output for the lab manual examples**
24. **A) Explaining the problem to your confused non-CS roommate**
25. **A) False - it's a historical record of your good intentions**

**Quiz - Paper 4: Answer Key**

1. **A) Prints "TU" (Pointer arithmetic moves the pointer ahead by 1 character)**
2. **A) 0 1 3 4 (continue skips i == 2)**
3. **B) 18 (y = 5, z = 7, x = x + y + z = 6 + 5 + 7 = 18)**
4. **A) Returns pointer to local variable that goes out of scope (Dangling pointer issue)**
5. **B) 11 (SQUARE(3+2) expands to 3+2\*3+2 = 3+6+2 = 11, due to macro expansion)**
6. **C) Undefined behavior (Buffer overflow, malloc(10) is too small)**
7. **A) calloc() initializes memory to zero; malloc() leaves it uninitialized**
8. **C) It's a trick question - all C programs eventually corrupt memory (😆 True in a way)**
9. **A) A pointer that refers to memory that has been freed**
10. **A) Use Valgrind or similar memory analysis tools**
11. **A) Inversely proportional: less sleep = more bugs = less sleep**
12. **D) You'll finish approximately 30 minutes after the deadline**
13. **B) Three - one to code, one to Google, one to pray**
14. **C) You frantically check if you submitted the right file**
15. **B) First-years don't know what pointers are; second-years wish they still didn't**
16. **D) All of the above, in that exact depressing order**
17. **D) Remove all your printf debugging statements because "it works now"**
18. **D) "seg fault (core dumped)" (😆 Classic)**
19. **D) Your sanity, one segmentation fault at a time**
20. **C) Heisenbug: the act of observing the code changes its behavior**
21. **A) AI learns from its mistakes**
22. **A) Segfaults per function**
23. **C) Use the rand() function but forget to seed it with srand()**
24. **A) Yes, but only until you need to modify it next week**
25. **A) You immediately get assigned to maintain it forever**

**Quiz - Paper 5: Answer key**

1. **A) 3 (In Python, True is treated as 1, so 2 + True = 2 + 1 = 3)**
2. **A) Converts an empty array to boolean true (In JavaScript, !![] converts an array to boolean, and non-empty arrays are truthy)**
3. **A) 12 (PHP treats "5" and "7" as numbers when using +)**
4. **A) <div> is a block element, <span> is an inline element**
5. **A) The highest-paid employee (It orders by salary in descending order and limits to one result)**
6. **A) The code is extremely good/impressive**
7. **A) It's incredibly clever, possibly overcomplicated**
8. **A) To criticize it indirectly or subtly**
9. **A) A demanding customer who escalates minor issues**
10. **A) Controlling who gets to participate based on arbitrary standards**
11. **A) Inversely proportional to the number of CS students present**
12. **A) Strong positive correlation: visits increase exponentially near deadline**
13. **D) All of the above**
14. **D) It never peaks, only increases asymptotically over time**
15. **D) Their ability to explain technical concepts without jargon**
16. **D) This is a theoretical question as perfect code doesn’t exist**
17. **D) The documentation becomes longer than the code itself**
18. **B) An infinite number, as they’ll never reach consensus**
19. **A) Document extensively with “DO NOT TOUCH” warnings**
20. **A) Directly proportional: the most important code is always the ugliest**
21. **D) Frequency of muttering "this should be working"**
22. **D) All of the above, in that order**
23. **C) When no one understands how it works but everyone's afraid to touch it**
24. **D) All of the above, depending on the day**
25. **A) They become permanent features**

**Quiz - Paper 6**

1. **A) 555**
2. **A) "number"**
3. **A) [1, 2, 3, 4]**
4. **A) They're compared by reference, not value**
5. **A) true**
6. **A) Release it to production/users**
7. **D) All of the above**
8. **A) Today I Learned**
9. **A) Spending excessive time on trivial details**
10. **A) Tangled, unstructured code that's hard to follow**
11. **D) All of the above**
12. **D) All of the above**
13. **C) They're done with the fun parts**
14. **A) Inversely proportional: the hardest bugs have the simplest fixes**
15. **D) All of the above are valid techniques**
16. **A) Yes, but their personal brand won't benefit**
17. **D) N+1, where N is the number needed to find that Stack Overflow answer again**
18. **D) Immediately after writing it, when you still have context**
19. **A) Their ability to explain complex concepts in simple terms**
20. **D) All of the above**
21. **D) All of the above, simultaneously**
22. **D) "It works on my machine"**
23. **A) They ask the most basic questions without embarrassment**
24. **B) How little you hate it six months later**
25. **C) Include documentation updates in definition of done**