**Quiz - Paper 5**

**Instructions**

**- This quiz contains 25 multiple-choice questions.**

**- Select the best answer for each question.**

**- Time allowed: 30 minutes.**

bleedecode bleedecode bleedecode bleedecode bleedecode bleedecode bleedecode bleedecode bleedecode bleedecode bleedecode bleedecode bleedecode bleedecode bleedecode bleedecode bleedecode bleedecode bleedecode bleedecode bleedecode bleedecode bleedecode bleedecode bleedecode bleedecode bleedecode bleedecode

**1. What does printf("%d", 'A' + 1); output in C?**

A) A1  
B) 66  
C) B  
D) Compiler explodes

**2. What's the result of sizeof(printf("Hello")) in C?**

A) 5 (because "Hello" has 5 letters)  
B) 0 (because printf doesn’t return anything useful)  
C) 4 (if you're lucky)  
D) Depends on how much the compiler likes you today

**3. What happens when you write int arr[10] = {0}; printf("%d", arr[99]);?**

A) It prints 0 because memory is generous.  
B) Segmentation fault, your computer cries.  
C) Undefined behavior, enjoy the mystery.  
D) Compiler glares but lets it slide.

**4. What's the output of char \*s = "Hello"; s[0] = 'Y'; printf("%s", s);?**

A) Yello  
B) Compiler error (string literals are sacred!)  
C) Undefined behavior, but let's pretend it's fine.  
D) It hacks into your neighbor’s WiFi.

**5. What happens if you free() a pointer twice?**

A) Compiler sends a strongly worded email.  
B) You summon the Segmentation Fault Demon.  
C) Your program runs fine (until it doesn't).  
D) The computer becomes self-aware.

**6. What does "rubber duck debugging" mean?**

A) Talking to an actual rubber duck about your code.  
B) Explaining your problem out loud and magically fixing it.  
C) A new agile methodology no one understands.  
D) Debugging while taking a bath.

**7. What's a "heisenbug"?**

A) A bug that disappears when you try to debug it.  
B) A bug that exists in two states: working and not working.  
C) Walter White's side hustle in programming.  
D) A quantum bit flipped a variable and broke your code.

**8. What does "deprecated" actually mean?**

A) "We don’t use it anymore, but good luck!"  
B) "It will break in a future update, surprise!"  
C) "Use this at your own risk. We warned you."  
D) All of the above.

**9. What’s the best way to center a div?**

A) Cry and use margin: auto.  
B) Stack Overflow it for the 500th time.  
C) Use flexbox and pretend you always knew it.  
D) Burn it all down and use tables.

**10. What happens if you fork() too many times in C?**

A) Your computer begs for mercy.  
B) Your OS bans you.  
C) You create infinite child processes and regret everything.  
D) You become a father of many lost processes.

**11. What’s the *real* purpose of daily standup meetings?**

A) To keep managers entertained.  
B) To announce that you're "working on it".  
C) To remind yourself what you were supposed to do.  
D) All of the above.

**12. When a developer says, "I'll fix it soon," what does that actually mean?**

A) "Maybe next sprint. Maybe never."  
B) "If nobody notices, it’s not a bug."  
C) "I'll fix it right after my coffee break. (Lies)"  
D) All of the above.

**13. How do developers estimate project timelines?**

A) By guessing wildly.  
B) By doubling the first guess and adding two months.  
C) By pretending they have a plan.  
D) By using ancient developer magic.

**14. What's the most reliable way to debug code?**

A) Staring at it until something feels wrong.  
B) Print statements everywhere.  
C) Asking the intern to check it.  
D) Turning the computer off and on again.

**15. What does "legacy code" actually mean?**

A) Code written so long ago, nobody dares touch it.  
B) Code that works, but nobody knows how.  
C) Code so terrifying, new hires run away.  
D) All of the above.

**16. What’s the worst thing you can say in a code review?**

A) "Why is this working?"  
B) "Let's rewrite everything in Rust."  
C) "This function has 500 lines, but it’s fine."  
D) "I found a small bug... in every file."

**17. What happens when you introduce a "temporary fix" in production?**

A) It becomes permanent.  
B) It breaks something unrelated.  
C) Future you cries.  
D) All of the above.

**18. What's the best way to document your code?**

A) Write self-documenting code (good luck).  
B) Add a comment that says // don't ask everywhere.  
C) Explain it to your rubber duck.  
D) Hope your future self understands.

**19. How do experienced developers solve impossible bugs?**

A) Sacrifice a coffee cup to the programming gods.  
B) Close the IDE and go for a walk.  
C) Convince themselves it’s a feature, not a bug.  
D) All of the above.

**20. What’s the best debugging tool?**

A) printf() spam.  
B) A friend who actually reads the error message.  
C) The one tool you refuse to learn.  
D) "Have you tried turning it off and on again?"

**21. What’s the true meaning of “this shouldn’t be happening” when debugging?**

A) "The universe is broken."  
B) "I forgot something, but I don’t know what."  
C) "It works on my machine, so it’s not my problem."  
D) All of the above.

**22. What happens immediately after fixing a critical bug?**

A) Two new bugs appear.  
B) The fix breaks something else.  
C) You get a "quick" feature request.  
D) All of the above.

**23. How can you identify the most experienced developer in the team?**

A) They ask the dumbest questions.  
B) They have the biggest mug of coffee.  
C) They sigh the loudest when looking at legacy code.  
D) They disappear when deadlines are near.

**24. What's the developer equivalent of "the dog ate my homework"?**

A) "The deployment pipeline failed."  
B) "Works on my machine."  
C) "I thought someone else fixed it."  
D) "It’s not a bug, it’s a feature."

**25. What's the best way to prevent documentation from going out of date?**

A) Never write any.  
B) Lock it in a vault and never touch it.  
C) Add a disclaimer: *"May not be accurate, proceed at your own risk."*  
D) Accept that it was outdated the moment you wrote it.