h) Random crashes – you are given a source code to test and it randomly crashes and it never crashes in the same place (you have attached a debugger and you find this). Explain what all you would suspect and how would you go about with isolating the cause. Bonus – The deeper you go into computer architecture and explain, better.

When dealing with random crashes in a source code that occurs unpredictably and doesn't consistently crash in the same place, several factors could be contributing to the issue. Here's a brief overview of the areas to investigate and the steps to isolate the cause:

Memory Issues:

Suspect: Uninitialized variables, buffer overflows, or memory leaks.

Isolation: Use tools like Valgrind or AddressSanitizer to detect memory-related problems. Check for issues like invalid memory access or corruption.

Concurrency and Thread Safety:

Suspect: Race conditions or thread-related problems.

Isolation: Analyze the code for shared resources, critical sections, and synchronization mechanisms. Use thread sanitizers to identify threading issues.

Input or Environmental Factors:

Suspect: Variability in input data or environmental conditions.

Isolation: Attempt to reproduce the crash with specific inputs or under particular environmental setups. Employ fuzz testing to explore different input scenarios.

Fault Tolerance:

Suspect: Lack of error handling or unexpected input handling.

Isolation: Review error-handling mechanisms. Introduce faults intentionally to observe how the system responds. Stress-test the application to evaluate its resilience.

Dynamic Debugging:

Suspect: Code that exhibits non-deterministic behavior.

Isolation: Use a debugger (e.g., GDB) to collect information about the state of the application when it crashes. Set breakpoints strategically to narrow down the location of the crash.

Dependency and Library Issues:

Suspect: Incompatibility or outdated versions of libraries.

Isolation: Verify that all dependencies are up to date and compatible. Check the documentation for known issues related to library versions.

Hardware and Architecture Considerations (Bonus):

Suspect: Low-level issues related to the computer architecture.

Isolation: Investigate memory corruption, concurrency issues in multi-core architectures, and potential hardware failures. Check for interrupt-related problems.

Community and Documentation:

Suspect: Lack of awareness regarding known issues or community discussions.

Isolation: Search forums and communities for discussions related to the programming language, framework, or tools. Review documentation for any reported issues.

By systematically addressing these areas, you can isolate the cause of random crashes. Remember to document your findings and collaborate with the development team or community if needed. Debugging can be an iterative process, so patience and a methodical approach are essential.

f) What is dot product and cross product? Explain use cases of where dot product is used and cross product is used in graphics environment. Add links to places where you studied this information and get back with the understanding.

The dot product and cross product are two fundamental operations in vector algebra that are widely used in computer graphics. Both operations involve multiplying two vectors, but they produce different results with distinct applications.

Dot Product:

The dot product, also known as the scalar product, is a mathematical operation that takes two vectors and returns a scalar quantity. It is represented by the symbol · or .. Geometrically, the dot product represents the projection of one vector onto another. The magnitude of the dot product is equal to the product of the magnitudes of the two vectors multiplied by the cosine of the angle between them.

Use Cases of Dot Product in Graphics:

Calculating the angle between two vectors: The dot product can be used to calculate the cosine of the angle between two vectors. This information is useful for various applications, such as determining the orientation of objects in a 3D scene or calculating the reflection of light off a surface.

Projecting a vector onto another vector: The dot product can be used to project a vector onto another vector. This is useful for tasks such as calculating shadows or creating animations.

Determining if two vectors are orthogonal: If the dot product of two vectors is zero, then the vectors are orthogonal, meaning they are perpendicular to each other. This property is useful for various applications, such as defining the normal vector of a plane or determining the direction of a force perpendicular to a surface.

Cross Product:

The cross product, also known as the vector product, is a mathematical operation that takes two vectors and returns a vector perpendicular to both of them. It is represented by the symbol × or ⨯. Geometrically, the cross product represents the area of a parallelogram formed by the two vectors. The magnitude of the cross product is equal to the product of the magnitudes of the two vectors multiplied by the sine of the angle between them.

Use Cases of Cross Product in Graphics:

Calculating the normal vector of a triangle: The cross product of two vectors lying on a plane is equal to the normal vector of the plane. This is useful for calculating the normal vector of a triangle, which is essential for shading and other lighting calculations.

Calculating the torque of a force: The cross product of a force vector and a position vector is equal to the torque of the force. This is useful for calculating the torque of a force on an object in a 3D scene.

Determining the direction of a right-hand turn: The cross product of two vectors points in the direction of a right-hand turn from the first vector to the second vector. This is useful for various applications, such as determining the rotation axis of an object or calculating the direction of a force perpendicular to a surface.

References:

Dot product: https://en.wikipedia.org/wiki/Dot_product

Cross product: https://en.wikipedia.org/wiki/Cross_product

Vector algebra: https://en.wikipedia.org/wiki/Vector_algebra

Computer graphics: https://en.wikipedia.org/wiki/Computer_graphics

g) Explain a piece of code that you wrote which you are proud of? If you have not written any

code, please write your favorite subject in engineering studies. We can go deep into that

Subject.

Artificial Intelligence (AI) is a testament to humanity's relentless pursuit of understanding and replicating the intricate processes of cognition. It transcends mere technology, evolving into a profound exploration of what it means to think, learn, and adapt. In the realm of AI, algorithms become the architects of intelligence, mirroring the intricate dance of neurons in the human brain. It is not merely about creating smart machines; it is about unraveling the mysteries of consciousness and pushing the boundaries of what is conceivable. As we forge ahead, AI is not just a field of study; it is a profound reflection on the essence of intelligence itself—a quest to breathe life into the inanimate and awaken a new form of synthetic sentience.