36 Lecture - CS501

Important Subjective

1. **What is floating-point arithmetic, and why is it commonly used in scientific and engineering applications?**
   Answer: Floating-point arithmetic is a method of performing mathematical calculations with real numbers that have both a whole number part and a fractional part. It is commonly used in scientific and engineering applications where high precision is required because it can represent a wide range of values and maintain accuracy even when dealing with very small or large numbers.

   What is the difference between single-precision and double-precision floating-point formats?
   Answer: Single-precision floating-point format uses 32 bits to represent a real number, while double-precision floating-point format uses 64 bits. This means that double-precision format can represent a larger range of values and has higher precision than single-precision format.

   What is a normalized floating-point number, and why is it useful?
   Answer: A normalized floating-point number is a number in which the most significant bit of the mantissa is always 1. This allows for a wider range of representable values and more precise calculations, as well as easier comparison and manipulation of numbers.

   What is the significance of the exponent in floating-point arithmetic?
   Answer: The exponent determines the magnitude of the number being represented and indicates the position of the decimal point. It also allows for efficient scaling of values and enables the representation of both very small and very large numbers.

   What is the difference between rounding and truncation in floating-point arithmetic?
   Answer: Rounding involves adjusting the result of a calculation to the nearest representable value, while truncation involves discarding the least significant bits of the result. Rounding is generally preferred for accuracy, while truncation is faster but may introduce errors.

   What are the most common floating-point exceptions, and how are they typically handled?
   Answer: The most common floating-point exceptions are overflow, underflow, and division by zero. They are typically handled by triggering an exception and either returning a special value or halting the program.

   How does the use of subnormal numbers affect the precision and accuracy of floating-point arithmetic?
   Answer: Subnormal numbers allow for the representation of very small values that would otherwise be lost due to rounding or truncation. However, their use can decrease precision and accuracy due to the increased number of bits required to represent them.

   What is the difference between a denormalized and normalized floating-point number?
   Answer: A normalized floating-point number has a nonzero mantissa and an exponent that is

adjusted to represent the value accurately, while a denormalized number has a zero exponent and a smaller range of representable values.

How does the use of floating-point arithmetic affect the performance of computer programs?
Answer: Floating-point arithmetic is generally slower than integer arithmetic due to the increased complexity of the operations and the larger number of bits required to represent real numbers. However, its use is often necessary in scientific and engineering applications where high precision is required.

What are some common techniques for improving the performance of floating-point arithmetic operations?
Answer: Some common techniques include using specialized hardware or software libraries for floating-point calculations, optimizing algorithms and data structures to minimize the number of operations required, and using parallel processing to distribute calculations across multiple processors.