Problem: Write a simple function in C that determines whether a system is big or little endian.

- First, can the candidate clearly define what endian refers to?
    - Endian refers to byte ordering within a word.
- Can the candidate define the difference between big and little endian?
    - A big-endian system stores the most significant byte of a word at the smallest memory address and the least significant byte at the largest.
- Can the candidate give a concrete example in words?
    - E.g. In a 32-bit system, the native word consists of 4 bytes. In a big-endian system, for the word 0x12345678, 0x12 will be stored in the lowest addressed byte, or byte 0.

```
bool isBigEndian(void) {
      uint32_t word = 0x1;
      uint8_t *byte_ptr = (uint8_t *)&word;
      return *byte_ptr == 1;
}
```

- Can the candidate define a reasonable API / function prototype?
    - Can be boolean or return an enum which indicates big or little endian
- Given the above verbal definition of endian and example, can they translate it into a working C function?
- Do they correctly use types?
- Do they correctly use a pointer?