Problem 1: C++ to MIPS

- Dealing with C++ `struct` types

- Only two kinds of data structures: arrays and linked-lists
  - Array pointer calculation: base + size*offset

- Complex if-conditions (`&&`, `||`, nested)

- For-loops, while-loops, do-while loops

- Argument and return-value register calling conventions

- No points off for: slow code, pseudo-instructions, minor syntax errors
Problem 2: Pipelines

- Data hazards, especially load-use hazards
- Control hazards, especially together with data hazards
- Stalls and flushes show up as \texttt{nop} instructions in pipeline
- What can be done to reduce/eliminate stalls?
- What can be done to reduce/eliminate flushes?
Problem 3: Cache Performance

- Hits and misses while processing data structure (array or linked list)
- Spatial and temporal locality
- Formulas:
  - Block-size
  - Associativity
  - Cache size
- Modified data structure/algorithm
Problem 4: Virtual Memory

- Why do we have virtual memory? (two reasons)

- What is the purpose of the page table?

- What is the purpose of the TLB?

- Why don’t we just store our data in the TLB?

- Virtual → Physical → Cache translation

- VM/cache connection:
  - Why don’t we use the cache instead of the TLB?
  - Why isn’t the cache accessed using virtual addresses?
Example 1: Search for item in an array

```c
bool exists(int *array, int n, int item) {
    for(int i = 0; i < n; ++i)
        if(array[i] == item)
            return true;
    return false;
}
```

Example 2: Count occurrences of item in array

```c
int count(int *array, int n, int item) {
    int c = 0;
    for(int i = 0; i < n; ++i)
        c += (array[i] == item);
    return c;
}
```