## Flynn Taxonomy

1. Explain SISD and give an example.

2. Explain SIMD and give an example.

3. Explain MISD and give an example.

4. Explain MIMD and give an example.

## Average Memory Access Time

AMAT stands for Average Memory Access Time. It refers to the time necessary to perform a memory access on average. It does NOT refer to the time necessary to execute an instruction which accesses memory. The AMAT of a simple system with only a single level of cache may be calculated as:

$$\text{AMAT} = (\text{hit time}) + (\text{miss rate}) * (\text{miss penalty})$$

This formula can be extended to more complicated memory hierarchies by replacing miss penalty with the AMAT for the next level in the memory hierarchy. Specifically for the $i^{th}$ level of memory:

$$\text{AMAT}_i = (\text{hit time})_i + (\text{miss rate})_i * (\text{miss penalty})_{i+1}$$

1. Calculate the AMAT for a system with the following properties:

   - L1$ hits in 1 cycle with local hit rate 50%
   - L2$ hits in 10 cycles with local hit rate 75%
   - L3$ hits in 100 cycles with local hit rate 90%
   - Main memory always hits in 1000 cycles

2. Calculate the AMAT for a system with the following properties:

   - L1$ hits in 1 cycle with global miss rate 50%
   - L2$ hits in 10 cycles with global miss rate 25%
   - L3$ hits in 100 cycles with global miss rate 10%
   - Main memory always hits in 1000 cycles

## Cache Performance

1. Suppose our processor has a separate L1 instruction cache and data cache. Our $\text{CPI}_{\text{ideal}}$ is 3 clock cycles, whereas memory access takes 125 cycles. Our I\$ miss rate is 4%, while our D\$ miss rate is 8%. Unfortunately, 60% of our instructions happen to be loads or stores. What is our processor's $\text{CPI}_{\text{stall}}$?

2. To improve the performance of our processor, we add a unified L2 cache between the two L1 caches and main memory. Our L2 cache has a hit time of 25 cycles and a global miss rate of 3%. What is our new $\text{CPI}_{\text{stall}}$?

## Cache Question

1. The following C function `sum_iter` is run on a 32-bit MIPS machine. On this system, textttstruct Nodes are aligned to two-word boundaries, since `sizeof(struct Node)` is 8. Assume the total space taken up by the linked list is greater than (and a multiple of) the cache size.

```
struct Node {
  int n;
  struct Node *next;
};
int sum_iter(struct Node *head) {
  int sum = 0;
  while (head != NULL) {
    sum += head - >n; // load from head+0
    head = head - >next; // load from head+4
  }
  return sum;
}
```

   Given a direct-mapped data cache with 13 index bits and 7 offset bits, how many words are in a cache block and how many bytes of data does this cache hold?

2. What is the lowest and highest possible cache hit rates for the `while` loop in `sum_iter`, and under what conditions do each occur?