7. (8 points) What is the difference between a *set associative* and *direct-mapped* cache, **briefly** describe a situation where a *set associative* cache with the *same capacity* performs better than a *direct-mapped* cache version.

Solution:

In a direct mapped cache every memory location can map to only one cache location. In some cases this can cause conflict misses even though there is in principle room in the cache. Set associative caches allow a memory location to be mapped to a set of locations (i.e. a 2-way set associative cache allows mapping to 2 locations) in the cache. This reduces conflict misses.

For example assume a cache with a capacity of 8 words, and consider the code below:

```
1 one:   lw $s1, 0($s0)  # first read
2 two:   lw $s2, 4($s0)  # second read
3 three: lw $s3,32($s0)  # third read
4 four:  lw $s4,36($s0)  # fourth read
5 five:  lw $s5, 0($s0)  # re-read first
6 six:   lw $s6, 4($s0)  # re-read second
```

If you use a direct mapped cache the addresses 0, 32 and 4, 36 will map to the same cache location. So the first two accesses will be compulsory misses, but will fill the cache location 0 and 1. Although we still have room in the cache (only 2 out of 8 is occupied), the next two reads (three and four) will again map to locations 0 and 1 overwriting the old ones. The last two reads will then again be cache misses.

In a 2-way set associative cache, the reads at three and four will not overwrite the old content because there is *another way* to store them in the cache. Therefore, the last two accesses (five and six) will come from the cache.