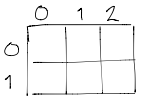
5.1)

```
for (i=0; i<8; i++)
   for (j=0; j<8000; j++)
      A[i][j] = B[j][0] + A[j][i];
```

5.1.1 How many 32 bit ints in 16 byte cache block?
4

5.1.2 Which variables exhibit temporal locality?
j, i

5.1.3 Spatial locality
C codes gets stored rows first.



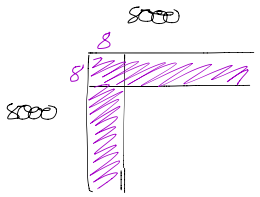
3×2 So we have spatial locality when accessing several items on the same row.

```
for i=1:8
  for j=1:8000
    A(i,j) = B(i,0) + A(j,i);
  end
end
```

5.1.4 How many 16-byte cache blocks are needed to store all 32-bit elems referenced?
$(8\times8000 + 8000 + 8000\times8 - 8\times8)\cdot\frac{1}{4} = 33984$

5.1.5 Temporal
i, j

5.1.6 Spatial
A(j,i), B(j,0)



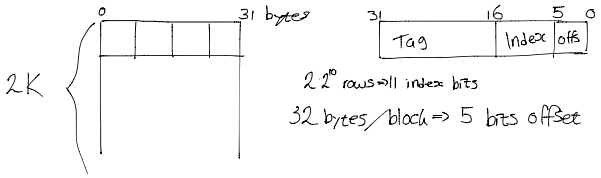
5.5)

512 KiB working set with address stream: 0 2 4 6 8 10 12 ... (512 KiB)

5.5.1 Assume 64 KiB direct mapped cache with 32-byte block. What is the MR? How is it sensitive to the set size or cache size?

$\frac{64}{32}$ = 2K blocks

1 bytes per elem
32 bytes per block



2^{11} rows → 11 index bits
32 bytes/block ⇒ 5 bits offset

First access ⇒ Miss, read 16 elems to cache (1, 3, 5, ... garbage)
17th access ⇒ —— 11 ——

Miss every 16th access. Only cold misses due to streams not having locality.

5.5.2 MR for 16-byte, 64 and 128-byte block size.

16 byte: 1/8
64 byte: 1/32
128 byte: 1/64

What kind of locality is exploited?
Spatial. We read sequentially.

5.5.3 Assume a two-entry stream buffer.
First read would give a miss, then no misses.
Cache block size (B) can affect both miss rate and miss latency. Assuming CPI=1 and an avg of 1.35 references per instruction find the optimal block size given the following MR for block sizes:

B:	8	16	32
	4%	3%	2%

5.5.4 What is the optimal block size for miss latency of $20 \times B$ cycles?

B=8: 0.04(8·20)
B=16: 0.03(16·20) 8 wins!
⋮

5.5.5 $24+B$ cycles. 24 units of time to fetch the first elem and the consecutive elems takes 1 unit.

B=8: 0.04(24+8)
B=8: 0.03(24+16)
⋮

5.5.6 For const miss latency?
Only the missrate matters!

5.11

5.11 As described in Section 5.7, virtual memory uses a page table to track the mapping of virtual addresses to physical addresses. This exercise shows how this table must be updated as addresses are accessed. The following data constitutes a stream of virtual addresses as seen on a system. Assume 4 KiB pages, a 4-entry fully associative TLB, and true LRU replacement. If pages must be brought in from disk, increment the next largest page number.

4669, 2227, 13916, 34587, 48870, 12608, 49225

TLB

Valid	Tag	Physical Page Number
1	11	12
1	7	4
1	3	6
0	4	9

Page table

Valid	Physical Page or in Disk
1	5
0	Disk
0	Disk
1	6
1	9
1	11
0	Disk
1	4
0	Disk
0	Disk
1	3
1	12

5.11.1 Show the final stage of the system given the above info. Also show if it's a hit in TLB Page table or page fault.

4 KiB pages $\Rightarrow 4 \cdot 2^{10} \Rightarrow$ 12 bits to encode

4669: 0001 0010 0011 1101 (12 bits)
Tag — Byte offset in page

Tag 1 is not in TLB $\Rightarrow$ TLB miss
(Tag = Virtual Page)

5.11.2 Use 16 KiB instead of pages. Advantages / Disadvantages?

+ Fewer misses since you load larger chunks.
- Larger miss penalty!