[3pt] Select all of the optimization blockers listed below

- ☐ mis-predicted branches
- ☐ data dependences
- ☐ memory aliasing
- ☐ pointer escaping
- ☐ procedure calls
- ☐ Chuck Norris

[3pt] Choose the **best** example of reduction in strength of the following expression `x = x * 32;`

- ☐ `x *= 32;`
- ☐ `x = (x << 2) << 3;`
- ☐ `x = (x << 3) << 2;`
- ☐ `x = x << 5;`
- ☐ `x <<= 5;`

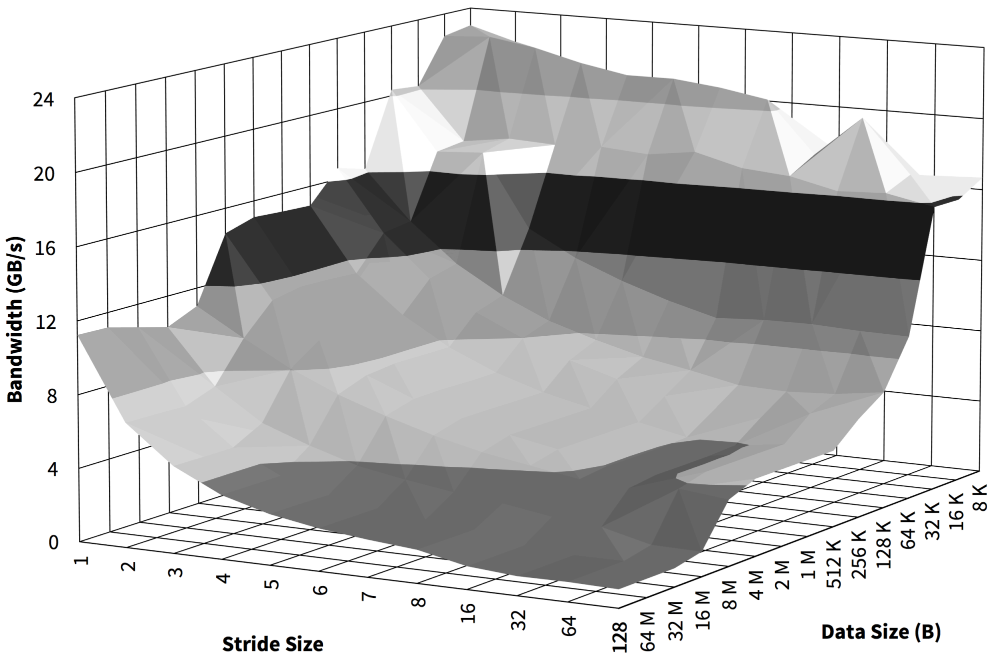
[3pt] Which is an example of a serial computation?

- ☐ `((((((1 * x0) * x1) * x2) * x3) * x4) * x5)`
- ☐ `(((1 * (x0 * x1)) * (x2 * x3)) * (x4 * x5))`
- ☐ `x0 * x1 * x2 * x3 * x4 * x5`
- ☐ `((((1 * x0) * x1) * x2) * (((1 * x3) * x4) * x5))`

[6pt] Name three "high-performance" optimizations. What do they do? You can focus on specific compiler optimizations or manual code modifications.

[8pt] What is the difference between `-march` and `-mtune`? Consider the instruction set architecture and code generation aspects of the compiler.

Given the following "memory mountain" diagram, please indicate the following.



(a) [4pt] Draw arrows pointing to the outermost cache and main memory ridges where stride=1

(b) [2pt] Which axis corresponds to spatial locality change? ☐ Data Size ☐ Stride Size

(c) [2pt] Which axis corresponds to temporal locality change? ☐ Data Size ☐ Stride Size

(d) [2pt] Estimate the size of the outermost cache (assume there are two arrays)

☐ 256KB

☐ 1MB

☐ 4MB

☐ 16MB

[6pt] Suppose we have a L1 cache miss penalty of 10 cycles and a L2 cache miss penalty of 100 cycles. What is the average access time if we have a miss rate of 10% in L1 cache and 2% in L2 cache? Assume a miss in L1 will be a hit in L2. You can just give the equation.

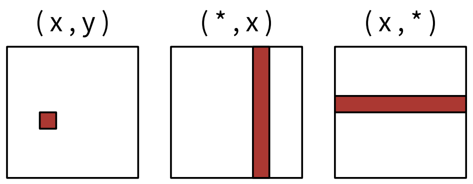
[8pt] Why is virtual memory addressing so important? Provide at least two reasons why. Give a use case with each reason.

The following question relates to matrix multiplication and how its access pattern can be improved.

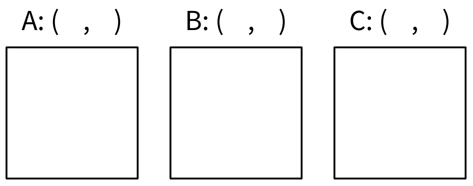
```
int A[N][N]; // input
int B[N][N]; // input
int C[N][N]; // output

for (i = 0; i < N; ++i) {
  for (j = 0; j < N; ++j) {
    for (k = 0; k < N; ++k) {
      C[i][j] += A[i][k] * B[k][j];
    }
  }
}
```

Sample Access Patterns

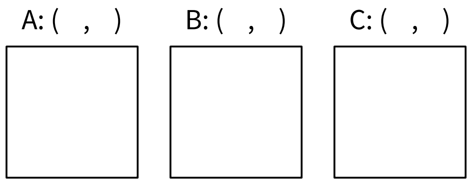


(a) [3pt] Diagram the current access pattern for A, B, and C. Please include variables indicating what changes with each inner iteration. the inner-loop variable can be abbreviated with * to indicate the access pattern movement. Sample access patterns are listed above.



(b) [3pt] Reorder the loops to improve the access pattern to be optimal (or near optimal). You can just list the variables at each increasing nest level. e.g. `i-j-k` for the default access pattern.

(c) [3pt] Diagram the new access pattern for A, B, and C



(d) [3pt] Name an additional technique that will improve the cache hit rate.

The following question related to cache configurations For each of the following cache configurations, please list the hit/miss list of the addresses listed at the top. Also show the contents of the cache at the end. Assume all cache entries are valid.

(a) [6pt] Final cache contents

(b) [6pt] Hit/Miss Record for each address

#	*Address*	*Binary*								*Cache #1 Hit?*	*Cache #2 Hit?*
1	`0x4C`										
2	`0x6F`										
3	`0xED`										

Cache Config #1 Addresses

Tag			*Set*		*Block*		

Cache Config #2 Addresses

	Tag		*Set*			*Block*		
#1								
#2								
#3								

Cache #1 Contents

Set	*LRU*	*Tag*			*Memory Block*
0	0	0	0	1	M[20-27]
	1	1	1	1	M[E0-E7]
1	1	0	1	0	M[48-4F]
	0	1	1	1	M[E8-EF]
2	1	1	1	0	M[D0-D7]
	0	1	0	1	M[B0-B7]
3	0	0	0	0	M[18-1F]
	1	0	1	0	M[58-5F]

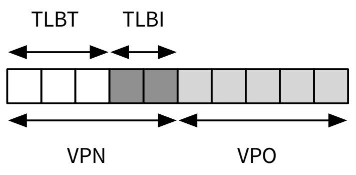
Cache #2 Contents

Set	*Tag*		*Memory Block*
0			
1	0	1	M[28-2F]
2	1	1	M[58-5F]
3	0	1	M[68-6F]
4	0	0	M[80-87]
5	1	1	M[B8-BF]
6	1	0	M[D0-D7]
7			

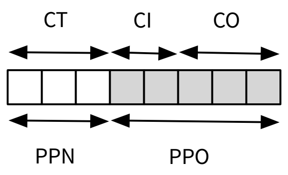
(c) [1pt] Which of the two cache configurations are better for this data? Why?

[12pt] Fill in the top two tables. If there is not a TLB hit, assume you can read the value from the page table. If there is not an entry in the page table, assume there is no mapping from the virtual address to physical address.

10-bit Virtual Address



8-bit Physical Address



Virt Addr	TLBT			TLBI		TLB Hit?
0x006						
0x1DA						
0x07F						

Virt Addr	Phys Addr	CT			CI		Cache Hit?
0x006							
0x1DA							
0x07F							

TLB Layout

Set	*Tag*			*Valid*	*PPN*
0	0	0	0	1	0x2
	0	0	1	1	0x0
1	0	1	1	1	0x3
				0	-
2	0	1	1	1	0x5
	0	1	0	1	0x6
3				0	-
				0	-

Cache Layout

Set	*Tag*			*Valid*	*Block*
0	0	0	1	1	M[20-27]
	1	1	1	1	M[E0-E7]
1	0	1	0	1	M[48-4F]
	1	1	1	1	M[E8-EF]
2	1	1	0	1	M[D0-D7]
	1	0	1	1	M[B0-B7]
3	0	0	0	1	M[18-1F]
	0	1	0	1	M[58-5F]

Page Table

VPN	*PPN*	*Valid*	*VPN*	*PPN*	*Valid*	*VPN*	*PPN*	*Valid*	*VPN*	*PPN*	*Valid*
0x00	0x2	1	0x08	-	0	0x10	-	0	0x18	-	0
0x01	-	0	0x09	-	0	0x11	-	0	0x19	-	0
0x02	-	0	0x0A	-	0	0x12	-	0	0x1A	-	0
0x03	0x7	1	0x0B	0x6	1	0x13	-	0	0x1B	-	0
0x04	0x0	1	0x0C	-	0	0x14	-	0	0x1C	0x4	1
0x05	-	0	0x0D	0x3	1	0x15	-	0	0x1D	-	0
0x06	-	0	0x0E	0x5	1	0x16	-	0	0x1E	-	0
0x07	-	0	0x0F	-	0	0x17	-	0	0x1F	0x1	1

(a) Virtualization

   i. [4pt] Under the concept of processor virtualization, what is a fundamental difference between gang scheduling and hypervisors?

(b) USB

   i. [2pt] (True/False) _______ USB 3.0 is backwards compatible with USB 2.0

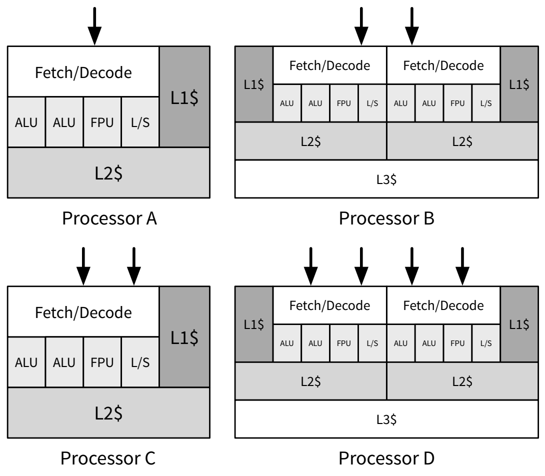
   ii. [4pt] List two differences between USB 2.0 and USB 3.0

(c) PCI Express (PCIe)

   i. [2pt] What is the format of all data transmitted over the PCI Express Bus?

   ii. [2pt] Devices are able to use varying amounts of bandwidth through PCI Express. They can vary from 1x to 32x. What is this basic unit called?

(a) Multicore and Simultaneous Multithreading



i. [2pt] Which processor(s) are multi-core?
☐ A ☐ B ☐ C ☐ D

ii. [2pt] Which processor(s) are hyperthreaded?
☐ A ☐ B ☐ C ☐ D

iii. [4pt] Why is L1 cache not coherent? What is the importance of cache coherence?

iv. [4pt] How does a snoopy cache work? Consider what is changed for cache lines and the protocol required for communication.

(b) SIMD and Vectorization

i. [4pt] Name at least three different iterations of SIMD advancements and describe what functionality was added. How do these new features help?

ii. [2pt] What are intrinsics? Why would a programmer use them?

iii. [6pt] Consider the following equation: $d = (a + b) * c + 10$. How would this be computed using the x87 FPU? How would this be computed using the SIMD scalar operations? Include memory operations where appropriate. **Hint:** x87 is stack-based.

(c) [2pt] ARM has a ______________ architecture while x86_64 has a ______________ architecture.

(d) [3pt] How does conditional execution work with the ARM ISA?

(e) [2pt] What are two configurable options in ARM?

(g) [4pt] Only using the barrel shifter, RSB (reverse subtract) and ADD, evaluate $r_2 = r_1 \times 33 \times 63$
Hint: use `LSL` for the barrel shifter