Name: _____ **CompID:** _____

For each question, either (a) mark one circle or (b) circle one of T or F for each row or (c) fill in the blank. Unanswered questions will be weighted slightly higher than incorrectly marked questions.

Feel free to write clarifying comments next to any question or option that needs them.

.....

Question 1 A DNS lookup on `kytos.cs.virginia.edu` results in the address `128.143.67.106`. An HTTPS request to `kytos.cs.virginia.edu` contains the address `128.143.67.106`

- ○ in plain-text in the TCP header
- ○ in plain-text in the IP header
- ○ in plain-text in the HTTP headers
- ○ encrypted in the TCP header
- ○ encrypted in the IP header
- ○ encrypted in the HTTP headers

Question 2 Adding pipelining to a processor in general

T F increases throughput

T F increases latency

Question 3 An optimistic transaction operates by

- ○ verifying that shared memory was not changed during its operation
- ○ using a private copy of memory instead of sharing memory
- ○ preventing other threads from accessing shared memory during its operation
- ○ passing messages via a concurrent queue instead of sharing memory

Question 4 Diffie-Hellman's security depends on both the existence of hard-to-invert functions on finite cyclic groups, and also on

- ○ two parties creating, and sharing only with one another, random numbers
- ○ two parties creating, and never sharing, random numbers
- ○ one party creating, and sharing only with one other party, a random number
- ○ one party creating, and never sharing, a random number

Question 5 Dynamic approaches to deadlock prevention work by analyzing each request for a resource and intervening if that request could result in deadlock. Static approaches instead analyze the source code prior to running to detect and remove potential deadlocks. Common threading libraries such as pthreads

- ○ Have static deadlock prevention enabled by default
- ○ Have static deadlock prevention but you have to enable it
- ○ Have neither static nor dynamic prevention
- ○ Have dynamic deadlock prevention enabled by default
- ○ Have dynamic deadlock prevention but you have to enable it

Information for Q6–Q8 Explain how the following code

```
addq %r8, %r9
addq %r9, %r10
```

interacts with an eight-stage pipeline where

- all instructions put values into the register file at the end of the eighth stage
- all instructions retrieve values from the register file at the beginning of the third stage
- `addq` places the sum into a pipeline register at the end of the fifth stage
- `addq` needs both operands to be available at the beginning of the fourth stage

and with additional constraints given in the questions below

Question 6 Assume all clock cycles (the period of time between two consecutive rising clock edges) are numbered, with 0 being the first cycle when the computer turns on, 1 the cycle after that, etc.
If the first `addq` enters the first stage on cycle 20,000 and both `%r8` and `%r9` have the right values in the register file before then, what will be the last cycle number when that `addq` is still somewhere in the pipeline?

Answer: _____

Question 7 Assuming the pipeline **does not forward** (that is, it reads values only from the register file and only in the appropriate stage), for how many cycles will the second `addq` stall?

Answer: _____

Question 8 Assuming the pipeline **forwards** (that is, it reads values from the register file or any suitable pipeline register in whatever stage needs them), for how many cycles will the second `addq` stall?

Answer: _____

Information for Q9–Q11 I (Prof Tychonievich) took a wrong turn at Hunter college yesterday:

1. Starting from a stairwell, my destination was in the office suite at the end of the hall
2. I didn't know if it was the suite at the north or south end of the hall
3. I picked north, entered that office suite, but it was not the right one
4. I went back to the stairwell
5. I picked south, entered that office suite, and found my destination

Question 9 Which part of this story is an example of **speculative execution**?

- ○ walking into the wrong office suite and back to the stairwell
- ○ picking south the second time
- ○ picking north initially
- ○ being delayed because of the wrong turn
- ○ arriving at my destination

Question 10 Which part of this story is an example of a **measurable side-channel**?

- ○ walking into the wrong office suite and back to the stairwell
- ○ returning to the stairwell
- ○ picking south the second time
- ○ picking north initially
- ○ being delayed because of the wrong turn
- ○ arriving at my destination

Question 11 Which part of this story is an example of an **ephemeral action**?

- ○ walking into the wrong office suite and back to the stairwell
- ○ picking south the second time
- ○ picking north initially
- ○ being delayed because of the wrong turn
- ○ arriving at my destination

Question 12 In the code below, `dothis` is executed by several threads, each with a different multiple-of-1000 argument. All of its locks and barriers (which are numbered) are unnecessary, and one or more of them will cause deadlock.
Assume all globals are properly initialized, that barriers are initialized to the number of threads, and that `isprime` uses only local variables.

```
pthread_mutex_t m1, m2;
pthread_barrier_t b1, b2;
char *primes;

void *dothis(void *arg) {
    int start = (int)arg;
    pthread_mutex_lock(&m1);            // 1
    pthread_barrier_wait(&b1);          // 2
    pthread_mutex_unlock(&m1);          // 3
    for(int i = start; i < start + 1000; i += 1) {
        if (isprime(i)) {
            pthread_barrier_wait(&b2);  // 4
            primes[i] = 1;
        } else {
            pthread_mutex_lock(&m2);    // 5
            primes[i] = 0;
            pthread_mutex_unlock(&m2);  // 6

        }
    }
    return NULL;
}
```

For each set of numbered lines listed below, mark T if removing that set of calls would remove all deadlocks from this code; mark F if at least one deadlock would remain.

T F remove 5 and 6

T F remove 4

T F remove 2, 5, and 6

T F remove 2 and 4

T F remove 2

T F remove 1, 3, and 4

T F remove 1, 3, 5, and 6

T F remove 1 and 3

Question 13 JavaScript allows you to provide *event handlers*, functions to be run when an event (such as a key press, mouse move, etc) occurs. Common implementations of JavaScript guarantee that even if many events occur at once, one of those handlers will run at a time and each will be allowed to complete running before the next begins. This means that JavaScript event handlers run

- ○ neither concurrently nor in parallel
- ○ in parallel but not concurrently
- ○ concurrently but not in parallel
- ○ concurrently and in parallel

Question 14 Meltdown and Specter differ in several ways, including

- ○ they use different side-channels
- ○ one reads normally inaccessible memory, the other does not
- ○ one depends on speculative execution, the other does not
- ○ one depends on branch misprediction, the other does not

Question 15 Pick the word or phrase below that fits best in the following sentence:
"My _____ enforces sequential consistency of memory accesses."

- ○ programming language
- ○ operating system
- ○ hardware
- ○ data structure
- ○ algorithm

Question 16 Suppose a page fault occurs and **does not** result in a signal being generated. As a result of running the faulting instruction and handling the fault,

- T F the program containing the faulting instruction is aborted
- T F the processor enters kernel mode to handle the fault
- T F the faulting instruction is run twice
- T F the contents of the page table changes
- T F the contents of the TLB changes
- T F the contents of the L2 cache changes

Question 17 Suppose we want to add a guest account to a computer using the regular user account mechanisms but limiting as many activities as possible. Which of the following can we do for this guest account?

T F Limit the amount of time their programs can run

T F Limit the amount of memory their programs can allocate

T F Block access to the network

T F Block access to some system files that other user accounts may access

T F Block access to some system calls that other user accounts may use

T F Block access to some assembly instructions that other user accounts may use

Question 18 Suppose you access a page of memory for the first time, and the page is allocated in a four-level page table. Assume that none of the intermediate pages used in the lookup have been used before, but that the first level has been used recently. How many entries are added to the TLB by this access? Answer as a base-10 integer.

Answer: _____

Question 19 Suppose you discover that a program is reading decryption keys stored in kernel memory using the same variant of the Meltdown vulnerability that we discussed in class. Which of the following changes, if implemented by itself without implementing and other option listed in this question, would be sufficient to prevent that from happening?

T F run the program as a user account with fewer privileges

T F have the kernel swap the pages containing those keys out of memory before returning control to the user

T F disable the cache, doing just direct RAM access instead

T F disable out-of-order execution, doing just in-order pipelined execution instead

T F disable branch prediction, stalling each jump until learning which instruction will run after it

Information for Q20–Q22 The 4-level page tables we discussed from Intel architectures used 48-bit virtual addresses, 4KB pages, and 8B page table entries.

Question 20 How many pages of user-accessible memory can be allocated using only the minimal number of page table pages?

Answer: _____

Question 21 If we changed the page size to 16KB instead and left the number of levels unchanged, how many bits would virtual addresses become?

Answer: _____

Question 22 What is the minimal number of page table pages needed to allocate one page of user-accessible memory using this page table?

Answer: _____

Information for Q23–Q24 The following ask about the same set of information stored by the kernel. All options listed are stored by the kernel. Assume that each process has several threads.

Question 23 Which of the following are stored one per **process**?

T F the owning user account

T F the PTBR

T F an array of program register contents

T F a table of open files

Question 24 Which of the following are stored one per **thread**?

T F the owning user account

T F the PTBR

T F an array of program register contents

T F a table of open files

Question 25 What is the overall size of a 4-way set-associative cache with a 10-bit set index and 7-bit block offset? Answer in power-of-two byte notation like 64MB or 128B

Answer: _____

Question 26 When a program dereferences a null pointer, which of the following happens first?

- ○ the signal table is read
- ○ the signal handler is run
- ○ the page table is read
- ○ the page handler is run
- ○ the exception table is read
- ○ the exception handler is run

Question 27 Which of the following accelerate programs that exhibit *spatial* locality?

- T F pre-fetching
- T F large number of lines per set
- T F large cache blocks
- T F caches with single-byte blocks, single-line sets, and no pre-fetching

Question 28 Which of the following accelerate programs that exhibit *temporal* locality?

- T F pre-fetching
- T F large number of lines per set
- T F large cache blocks
- T F caches with single-byte blocks, single-line sets, and no pre-fetching

Question 29 Which of the following are **required** for deadlock to occur, such that if it is missing there can be no deadlock?

- T F Threads cannot steal resources from one another
- T F Threads can wait for a resource while maintaining access to other resources
- T F Threads can be blocked from acquiring a resource because too many other threads can access it
- T F There can be cycles in the resource allocation graph
- T F There are multiple threads, processes, or the like
- T F There are multiple resources

Question 30 Which of the following can occur without an exception and without exploiting a security flaw? Assume that `mst3k` is a regular user account.

T F kernel mode becomes `root`

T F kernel mode becomes `mst3k`

T F `root` enters kernel mode

T F `root` becomes `mst3k`

T F `mst3k` enters kernel mode

T F `mst3k` becomes `root`

Question 31 Which of the following permit *collisions*: that is, two inputs might give the same output without changing the algorithm or key(s) used.

T F symmetric-key encryption, like AES

T F public-key encryption, like RSA

T F private-key encryption, like RSA

T F cryptographic hashes, like SHA-2

Question 32 Your computer makes a DHCP request to a server to obtain

○ your computer's port

○ your computer's URL

○ your computer's IP address

○ the server's port

○ the server's URL

○ the server's IP address

○ a different computer's port

○ a different computer's URL

○ a different computer's IP address

. . . . . .

Pledge On my honor as a student, I have neither given nor received assistance on this exam.

{Signature}