Question 1. The register number that is used for the stack pointer (sp) in OS/161 is $29. It was found in the file kern/arch/mips/include/kern/regdefs.h

Question 2. OS/161 supports LAMEbus. It was found in kern/arch/sys161/include/bus.h

Question 3. The maximum number of CPU's that can be configured in OS/161 is 32. It was found in file kern/arch/sys161/include/maxcpus.h

Question 4. Since synchprobs is defined, hz = 10000.

Question 5. Since synchprobs is not defined, hz = 100

Question 6. If you wanted the debug statement to show, set the "dbflags" to match with the "d" argument. If you didn't want the debug statement to show, set the "dbflags" to something different than "d".

Question 7. To add and control a new set of debugging statements, add
#define DB_CATMOUSE 0x1000 to the list of the debug control statements

Question 8. DEBUG(DB_CATMOUSE){
Print("Hello World\n");
}//I would also have to #define DB_CATMOUSE

Question 9. (dbflags & DB_CATMOUSE) | (dbflags & DB_THREADS)

Question 10. The lock_acquire function is within the kprintf function. So by calling the debug statements will call itself.

Question 11. A bitmap is a struct that has an unsigned number and an unsigned char. It is usually used for storage management. It is an array of bits that is fixed in size. For example, if I wanted to have something like a file storage system, I would create a size 100bit bitmap using bitmap_create and bitmap_alloc.

Question 12. There are 4 states that a thread can be in: S_ZOMBIE, S_SLEEP, S_RUN,S_READY

Question 13. They get cleaned up in the exorcise function in thread.c

Question 14. The wchan_sleep function is used to put a thread to sleep

Question 15. The purpose of curthread is to keep track of the current thread

Question 16. Creates a thread fork on the current thread running on the cpu. It will create the thread fork 8 times calling add_thread 8 times. Next it will create the thread fork 8 times calling sub_thread 8 times. Eventually we expect the output to be back to the start value. This test is made to check if the final value is returned to the start value.

Question 17. Run 1: -18629
Run 2: -1775
Run 3: 4024
Run 4: 2392
Run 5: -43573

Question 18.

Question 19. Run 1: -6085
Run 2: -354
Run 3: 16606
Run 4: 4025
Run 5: -3064

Question 20. Multiple threads are trying to access “TEST_VALUE” at the same time. So “TEST_VALUE” is getting changed by multiple threads hence returning garbage values.