Started on	Monday, 24 January 2022, 7:09:07 PM
State	Finished
Completed on	Monday, 24 January 2022, 7:33:19 PM
Time taken	24 mins 12 secs
Grade	**10.61** out of 20.00 (**53**%)

Question **1**

Complete

Mark 1.00 out of 1.00

Why should a program exist in memory before it starts executing ?

- a. Because the variables of the program are stored in memory
- b. Because the memory is volatile
- [✓ c. Becase the processor can run instructions and access data only from memory]
- d. Because the hard disk is a slow medium

The correct answer is: Becase the processor can run instructions and access data only from memory

Question **2**

Complete

Mark 0.67 out of 2.00

Which of the following instructions should be privileged?

Select one or more:

- [ ] a. Access a general purpose register
- [x] b. Access I/O device.
- [ ] c. Access memory management unit of the processor
- [x] d. Set value of timer.
- [ ] e. Read the clock.
- [x] f. Modify entries in device-status table
- [x] g. Turn off interrupts.
- [x] h. Set value of a memory location
- [ ] i. Switch from user to kernel mode.

The correct answers are: Set value of timer., Access memory management unit of the processor, Turn off interrupts., Modify entries in device-status table, Access I/O device., Switch from user to kernel mode.

Question **3**

Complete

Mark 0.00 out of 2.00

Select all statements that correctly explain the use/purpose of system calls.

Select one or more:

- [✓ a. Switch from user mode to kernel mode]
- b. Run each instruction of an application program
- c. Provide services for accessing files
- d. Provide an environment for process creation
- e. Allow I/O device access to user processes
- [✓ f. Handle ALL types of interrupts]
- [✓ g. Handle exceptions like division by zero]

The correct answers are: Switch from user mode to kernel mode, Provide services for accessing files, Allow I/O device access to user processes, Provide an environment for process creation

Question **4**

Complete

Mark 0.80 out of 1.00

Select all the correct statements about two modes of CPU operation

Select one or more:

- [✓ a. Some instructions are allowed to run only in user mode, while all instructions can run in kernel mode]
- [✓ b. The two modes are essential for a multitasking system]
- [✓ c. The software interrupt instructions change the mode from user mode to kernel mode and jumps to predefined location simultaneously]
- d. The two modes are essential for a multiprogramming system
- [✓ e. There is an instruction like 'iret' to return from kernel mode to user mode]

The correct answers are: The two modes are essential for a multiprogramming system, The two modes are essential for a multitasking system, There is an instruction like 'iret' to return from kernel mode to user mode, The software interrupt instructions change the mode from user mode to kernel mode and jumps to predefined location simultaneously, Some instructions are allowed to run only in user mode, while all instructions can run in kernel mode

Question **5**

Complete

Mark 0.50 out of 0.50

Order the following events in boot process (from 1 onwards)

Shell	6
OS	3
Boot loader	2
BIOS	1
Init	4
Login interface	5

The correct answer is: Shell → 6, OS → 3, Boot loader → 2, BIOS → 1, Init → 4, Login interface → 5

Question **6**

Complete

Mark 1.00 out of 1.00

Match the register with the segment used with it.

esi	ds
eip	cs
edi	es
esp	ss
ebp	ss

The correct answer is: esi → ds, eip → cs, edi → es, esp → ss, ebp → ss

Question **7**

Complete

Mark 1.00 out of 1.00

```
int value = 5;
int main()
{
    pid_t pid;
    pid = fork();
    if (pid == 0) { /* child process */
        value += 15;
        return 0;
    }
    else if (pid > 0) { /* parent process */
        wait(NULL);
        printf("%d", value); /* LINE A */
    }
    return 0;
}
```

What's the value printed here at LINE A?

Answer: 5

The correct answer is: 5

Question **8**

Complete

Mark 1.00 out of 1.00

How does the distinction between kernel mode and user mode function as a rudimentary form of protection (security) ?

Select one:

[✓ a. It prohibits a user mode process from running privileged instructions]

b. It prohibits one process from accessing other process's memory

c. It disallows hardware interrupts when a process is running

d. It prohibits invocation of kernel code completely, if a user program is running

The correct answer is: It prohibits a user mode process from running privileged instructions

Question **9**

Complete

Mark 0.00 out of 2.00

xv6.img: bootblock kernel

```
    dd if=/dev/zero of=xv6.img count=10000
    dd if=bootblock of=xv6.img conv=notrunc
    dd if=kernel of=xv6.img seek=1 conv=notrunc
```

Consider above lines from the Makefile. Which of the following is incorrect?

- ✓ a. The size of the kernel file is nearly 5 MB
- ☐ b. The size of xv6.img is exactly = (size of bootblock) + (size of kernel)
- ☐ c. The bootblock may be 512 bytes or less (looking at the Makefile instruction)
- ☐ d. Blocks in xv6.img after kernel may be all zeroes.
- ✓ e. The xv6.img is of the size 10,000 blocks of 512 bytes each and occupies 10,000 blocks on the disk.
- ☐ f. xv6.img is the virtual processor used by the qemu emulator
- ☐ g. The size of the xv6.img is nearly 5 MB
- ☐ h. The xv6.img is of the size 10,000 blocks of 512 bytes each and occupies upto 10,000 blocks on the disk.
- ☐ i. The bootblock is located on block-0 of the xv6.img
- ✓ j. The kernel is located at block-1 of the xv6.img
- ☐ k. The xv6.img is the virtual disk that is created by combining the bootblock and the kernel file.

The correct answers are: xv6.img is the virtual processor used by the qemu emulator, The xv6.img is of the size 10,000 blocks of 512 bytes each and occupies upto 10,000 blocks on the disk., The size of the kernel file is nearly 5 MB, The size of xv6.img is exactly = (size of bootblock) + (size of kernel)

Question **10**

Complete

Mark 1.00 out of 1.00

Rank the following storage systems from slowest (first) to fastest(last)

You can drag and drop the items below/above each other.

- Magnetic tapes
- Optical disk
- Hard-disk drives
- Nonvolatile memory
- Main memory
- Cache
- Registers

Question **11**

Complete

Mark 0.14 out of 2.00

Select all the correct statements about calling convention on x86 32-bit.

- ✓ a. Space for local variables is allocated by substracting the stack pointer inside the code of the called function
- ☐ b. The ebp pointers saved on the stack constitute a chain of activation records
- ✓ c. Parameters may be passed in registers or on stack
- ☐ d. The two lines in the beginning of each function, "push %ebp; mov %esp, %ebp", create space for local variables
- ✓ e. The return value is either stored on the stack or returned in the eax register
- ☐ f. Compiler may allocate more memory on stack than needed
- ☐ g. Space for local variables is allocated by substracting the stack pointer inside the code of the caller function
- ✓ h. Paramters are pushed on the stack in left-right order
- ✓ i. Parameters may be passed in registers or on stack
- ✓ j. Return address is one location above the ebp
- ☐ k. during execution of a function, ebp is pointing to the old ebp

The correct answers are: Compiler may allocate more memory on stack than needed, Parameters may be passed in registers or on stack, Parameters may be passed in registers or on stack, Return address is one location above the ebp, during execution of a function, ebp is pointing to the old ebp, Space for local variables is allocated by substracting the stack pointer inside the code of the called function, The ebp pointers saved on the stack constitute a chain of activation records

Question **12**

Complete

Mark 0.50 out of 0.50

Compare multiprogramming with multitasking

● a. A multiprogramming system is not necessarily multitasking

○ b. A multitasking system is not necessarily multiprogramming

The correct answer is: A multiprogramming system is not necessarily multitasking

Question **13**

Complete

Mark 0.00 out of 2.00

Which of the following are NOT a part of job of a typical compiler?

- [✓ a. Invoke the linker to link the function calls with their code, extern globals with their declaration]
- b. Check the program for logical errors
- c. Suggest alternative pieces of code that can be written
- d. Convert high level langauge code to machine code
- [✓ e. Process the # directives in a C program]
- f. Check the program for syntactical errors

The correct answers are: Check the program for logical errors, Suggest alternative pieces of code that can be written

Question **14**

Complete

Mark 2.00 out of 2.00

Match the program with it's output (ignore newlines in the output. Just focus on the count of the number of 'hi')

main() { fork(); execl("/usr/bin/echo", "/usr/bin/echo", "hi\n", NULL); }	hi hi
main() { int i = NULL; fork(); printf("hi\n"); }	hi hi
main() { int i = fork(); if(i == 0) execl("/usr/bin/echo", "/usr/bin/echo", "hi\n", NULL); }	hi
main() { execl("/usr/bin/echo", "/usr/bin/echo", "hi\n", NULL); }	hi

The correct answer is: main() { fork(); execl("/usr/bin/echo", "/usr/bin/echo", "hi\n", NULL); } → hi hi, main() { int i = NULL; fork(); printf("hi\n"); } → hi hi, main() { int i = fork(); if(i == 0) execl("/usr/bin/echo", "/usr/bin/echo", "hi\n", NULL); } → hi, main() { execl("/usr/bin/echo", "/usr/bin/echo", "hi\n", NULL); } → hi

Question **15**

Complete

Mark 0.50 out of 0.50

Is the command "cat README > done &" possible on xv6? (Note the & in the end)

[✓ a. yes]

b. no

The correct answer is: yes

Question **16**

Complete

Mark 0.50 out of 0.50

Is the terminal a part of the kernel on GNU/Linux systems?

[✓ a. no]

b. yes

The correct answer is: no

◄ (Task) Compulsory xv6 task

Jump to...

(Optional Assignment) Shell Programming(Conformance tests) ►