Quiz

1.D 2.B 3.false 4.A 5.A 6.B 7.C 8.D 9. 2 2 4

10. A. Physical address of PDE: 0x0045d9fc

Physcal address of PTE: 0x0df2a4a0

FAILURE: The physical address of the table entry causing the failure is 0x0df2a4a0

B. TLB Hit: Physical address is 0x98f8a2c0

C. Physical address of PDE: 0x0045d0a0

Physical address of PTE: 0x000c3cbc

SUCCESS: The physical address accessed is 0x34abdcd0

1.Which of the following system calls can fail due to a network failure?
(a) socket(...) (b) listen(...) (c) bind(...) (d) gethostbyname(...)

2.Which of the following are copied on fork and preserved on exec?

(a) Global variables. (b) File descriptor tables. (c) Open file entry structs. (d) None of the above.

3.True/False: When requested to send 20 bytes over a network socket, execution will block until all 20 bytes have been sent. (a) True (b) False

4.Which of the following is a difference between blocking and ignoring a signal?

(a) Once a blocked signal is unblocked, it will be handled by the process. A signal that comes while it is being ignored will never be handled.

(b) SIGSTOP and SIGINT can be ignored, but not blocked.

(c) Ignoring a signal only causes it to have no effect, while blocking a signal returns the signal to its sender.

(d) None of the above

5. Simply decreasing the size of block headers used internally by malloc:
(a) Decreases internal fragmentation (b) Increases internal fragmentation (c) Decreases external fragmentation (d) Increases external fragmentation

6.On a 64 bit system, which of the following C expressions is equivalent to the C expression (x[2] + 4)[3]? Assume x is declared as int **x.

(a) *((*(x + 16)) + 28) (b) *((*(x + 2)) + 7)

(c) **(x + 28) (d) *(((*x) + 2) + 7) (e) (**(x + 2) + 7)

7. A program blocks SIGCHLD and SIGUSR1. It is then sent a SIGCHLD, a SIGUSR1, and another SIGCHLD, in that order. What signals does the program receive after it unblocks both of those signals (you may assume the program does not receive any more signals after)?

(a) None, since the signals were blocked they are all discarded.

(b) Just a single SIGCHLD, since all subsequent signals are discarded.

(c) Just a single SIGCHLD and a single SIGUSR1, since the extra SIGCHLD is discarded.

(d) All 3 signals, since no signals are discarded.

8. Assuming all the system calls succeed, which of the following pieces of code will print the word "Hello" to stdout?

(a) int fd = open("hoola.txt", O_RDWR); dup2(fd, STDOUT_FILENO); printf("Hello"); fflush(stdout);

(b) int fd = open("hoola.txt", O_RDWR); dup2(fd, STDOUT_FILENO); write(STDOUT_FILENO, "Hello", 5);

(c) int fd = open("hoola.txt", O_RDWR); dup2(fd, STDOUT_FILENO); printf("Hello");

(d) int fd = open("hoola.txt", O_RDWR); dup2(STDOUT_FILENO, fd); write(fd, "Hello", 5);

(e) int fd = open("hoola.txt", O_RDWR); dup2(fd, STDOUT_FILENO); write(fd, "Hello", 5);

9. For each code segment below, give the largest value that could be printed to stdout. Remember that when the system executes a signal handler, it blocks signals of the type currently being handled (and no others).

```
/* Version A */

int i = 0;

void handler(int s) {

        if (!i) {

                kill(getpid(), SIGINT);

        }

        i++;

}

int main() {

        signal(SIGINT, handler);

        kill(getpid(), SIGINT);

        printf("%d\n", i);

        return 0;
}
```

(1).Largest value for version A:_____

```
/* Version B */
int i = 0;
void handler(int s) {
    if (!i) {
        kill(getpid(), SIGINT);
        kill(getpid(), SIGINT);
    }
    i++;
}
int main() {
    signal(SIGINT, handler);
    kill(getpid(), SIGINT);
    printf("%d\n", i);
    return 0;
}
```

(2). Largest value for version B: _____

```
/* Version C */
int i = 0;
```

```
void handler(int s) {
    if (!i) {
        kill(getpid(), SIGINT);
        kill(getpid(), SIGUSR1);
    }
    i++;
}
int main() {
    signal(SIGINT, handler);
    signal(SIGUSR1, handler);
    kill(getpid(), SIGUSR1);
    printf("%d\n", i);
    return 0;
}
```

(3). Largest value for version C: _____

10. Address translation.

The contents of the relevant sections of memory are shown on this page. All numbers are given in hex- adecimal. Any memory not shown can be assumed to be zero. The Page Directory Base Address is 0x0045d000.

For each of the following problems, perform the virtual to physical address translation. If an error occurs at any point in the address translation process that would prevent the system from performing the lookup, then indicate this by circling FAILURE and noting the physical address of the table entry that caused the failure.

For example, if you were to detect that the present bit in the PDE is set to zero, then you would leave the PTE address in (b) empty, and circle FAILURE in (c), noting the physical address of the offending PDE.

1. Read from virtual address 0x9fd28c10.

(a) (TLB Hit) Physical address is: OR

(b) Physical address of PDE:

(c) Physical address of PTE:

(d) (SUCCESS) The physical address accessed is: OR (FAILURE) The physical address of the table entry causing the failure is:

2.Read from virtual address 0x0d4182c0.

(a) (TLB Hit) Physical address is: OR

(b) Physical address of PDE:

(c) Physical address of PTE:

(d) (SUCCESS) The physical address accessed is: OR (FAILURE) The physical address of the table entry causing the failure is:

3. Read from virtual address 0x0a32fcd0.

(a) (TLB Hit) Physical address is: OR

(b) Physical address of PDE:

(c) Physical address of PTE:

(d) (SUCCESS) The physical address accessed is: OR (FAILURE) The physical address of the table entry causing the failure is:

TLB			
Index	Tag	Frame Number	Valid
0	0x03506	0x98f8a	1
	0x27f4a	0x34abe	0
1	0x1f7ee	0x95cbc	0
	0x2a064	0x72954	1
2	0x1f7f0	0x95ede	0
	0x2005d	0xaa402	0
3	0x3fc2e	0x2029e	1
	0x3df82	0xff644	0

Address	Contents
000c3020	345ab236
000c3080	345ab237
000c332f	08e4523f
000c3400	93c2ed98
000c3cbc	34abd237
000c3ff0	93c2ed99
000c4020	8e56e237
000c432f	33345237
000c4400	43457292
000c4cbc	385ed293
000c4ff0	c3726292
0045d000	000c3292
0045d028	000c4297
0045d032	0df2a292
0045d0a0	000c3297
0045d3ff	0df2a236
0045d9fc	0df2a237
0df2a000	deded000
0df2a080	bc3de239
0df2a3fc	000c4296
0df2a4a0	00324236
0df2a4fc	df72c9a6
0df2b080	01f008c3
0df2bff0	000c5112