

10

20

Question 1

Not yet answered

Marked out of 5.00

Flag question

signal handler is a function that is executed when a process received an interrupt

Select one:

☐ True

☒ False



PREVIOUS ACTIVITY
CENG460_Quiz_3

Jump to...



Question 2

Not yet answered

Marked out of 5.00

Flag question

The following two processes, P1 and P2 are executed concurrently in the operating system. They share the semaphore variables S and R (each initialized to 1) and integer variable x (initialized to 0).

The concurrent execution of P1 and P2 result in one or both will be blocked forever?

```
void P1() {  
    while(1) {  
        semWait(S);  
        semWait(R);  
        x++;  
        semSignal(S);  
        SemSignal(R);  
    }  
}
```

```
void P2() {  
    while(1) {  
        semWait(R);  
        semWait(S);  
        x--;  
        semSignal(S);  
        SemSignal(R);  
    }  
}
```

Select one:

☒ True

☐ False

Next page

	9	10
8	19	20

Question 3

Not yet answered

Marked out of 5.00

Flag question

A thread that is blocked on a semaphore is awakened when another thread:

Select one:

- ☐ a. Tries to decrement a semaphore's value below 0
- ☒ b. Causes the semaphore's value to reach a specific number
- ☐ c. Tries to increment the semaphore
- ☐ d. Tries to block on the same semaphore.

[Clear my choice](#)



PREVIOUS ACTIVITY
CENG460_Quiz_3

Jump to...



Question 4

Not yet answered

Marked out of 5.00

Flag question

The fairest removal policy is first-in-first-out (FIFO): The process that has been blocked the longest is released from the queue first. A semaphore whose definition includes this policy is called a weak semaphore

Select one:

☐ True

☒ False

Next page



PREVIOUS ACTIVITY
CENG460_Quiz_3

Jump to...



Question 5

Not yet answered

Marked out of 5.00

Flag question

Starvation happens when a process can make no progress even if it is scheduled because there is a circular dependency on resources coupled with exclusive (locked) access

Select one:

☒ True

☐ False

Next page



PREVIOUS ACTIVITY
CENG460_Quiz_3

Jump to...



Question 6

Not yet answered

Marked out of 5.00

11: Flag question

The critical section is a section of code within a process that requires access to shared resources and that must not be executed while another process is in a corresponding section of code

Select one:

☒ True

☐ False

Next page



PREVIOUS ACTIVITY
CENG460_Quiz_3

Jump to...



Question 7

Not yet answered

Marked out of 5.00

Flag question

Consider the following code that creates two threads rather than the main thread. What is `pthread_join()` actually doing here?

```
#include <stdio.h>
#include <pthread.h>
// Code for thread A.
void *threadA(void *parameter)
{
    printf("Hello from Thread A with ID %u\n", (unsigned int)pthread_self());
    return NULL;
}
// Code for thread B.
void *threadB(void *parameter)
{
    printf("Hello from Thread B with ID %u\n", (unsigned int)pthread_self());
    return NULL;
}
int main()
{
    // create the two threads.
    pthread_t a,b;
    pthread_create(&a, NULL, &threadA, NULL);
    pthread_create(&b, NULL, &threadB, NULL);
    pthread_join(a, NULL);
    pthread_join(b, NULL);
    return 0;
}
```



```
pthread_join(a, NULL);  
pthread_join(b, NULL);  
return 0;  
}
```

Select one:

- ☐ a. Join wait for both threads to terminate but it is optional
- ☐ b. Join wait for both threads to terminate otherwise an error occurs
- ☒ c. Non of the above
- ☐ d. It is used to let A finish before B

[Clear my choice](#)

Question 8

Not yet
answered

Marked out of
5.00

Flag
question

Termination of the process terminates

Select one:

- ☐ a. First two threads of the process
- ☐ b. No thread within the process
- ☐ c. First thread of the process
- ☒ d. All threads within the process

[Clear my choice](#)



PREVIOUS ACTIVITY
CENG460_Quiz_3

Question 9

Not yet answered

Marked out of 5.00

Flag question

Threads are more memory-constrained due to OS limitation of the address space size of a single process

Select one:

☒ True

☐ False



PREVIOUS ACTIVITY
CENG460_Quiz_3

Jump to...



Question 10

Not yet answered

Marked out of 5.00

Flag question

No Need to any synchronization between threads

Select one:

☐ True

☒ False



PREVIOUS ACTIVITY
CENG460_Quiz_3

Jump to...



Question 11

Not yet
answered

Marked out of
5.00

Flag
question

If one thread fails, then the process fails

Select one:

☐ True

☒ False



PREVIOUS ACTIVITY
CENG460_Quiz_3

Jump to...



Question 12

Not yet
answered

Marked out of
5.00

Flag
question

If one thread opens a file with read privileges then

Select one:

- ☐ a. Other threads in the another process can also read from that file
- ☐ b. All of the mentioned
- ☒ c. Other threads in the same process can also read from that file
- ☐ d. Any other thread cannot read from that file

[Clear my choice](#)



PREVIOUS ACTIVITY
CENG460_Quiz_3

Question 13

Not yet
answered

Marked out of
5.00

Flag
question

When a process terminates, it is immediately removed from the system and all its parts will be removed from the memory

Select one:

☐ True

☒ False



PREVIOUS ACTIVITY
CENG460_Quiz_3

Jump to...



Question 14

Not yet
answered

Marked out of
5.00

Flag
question

The alarm() function sends the signal to the process that invoked the function when an alarm timing starts

Select one:

☐ True

☒ False



PREVIOUS ACTIVITY
CENG460_Quiz_3

Jump to...



8	9	10
18	19	20

Question 15

Not yet answered

Marked out of 5.00

Flag question

What is the output for the following code:

```
#include #include #include /*for signal()*/
void handler_3(int signum)
{ printf("Don't you dare shoot me one more time!\n");
  signal(SIGINT, SIG_DFL); }

void handler_2(int signum){
  printf("Hey, you shot me again!\n");
  signal(SIGINT, handler_3); }

void handler_1(int signum){
  printf("You shot me!\n");
  signal(SIGINT, handler_2); }

int main(){
  signal(SIGINT, handler_1);
  //loop forever! while(1); }
```

Select one:

- ☐ a. Don't you dare shoot me one more time!
- ☐ b. Hey, you shot me again!
- ☐ c. You shot me!
- ☐ d. none of the above

Clear my choice

tems (N ...

Question 16

Not yet
answered

Marked out of
5.00

Flag
question

The raise() function is a simple way for a process to send a signal to itself.

Select one:

☒ True

☐ False



PREVIOUS ACTIVITY
CENG460_Quiz_3

Jump to...



8	9	10
18	19	20

Question 17

Not yet answered

Marked out of 5.00

Flag question

What is the output of the following code:

```
void f(int signum)
{ printf("Ding!\n"); }

main()
{
    int i;
    i=fork();
    if(i !=0){
        signal(SIGALRM,f);
        kill(i,SIGALRM);
        pause();
        printf("MARHABA, I am awake!\n");
    }
    else {
        pause();
        printf("HELLO, I am awake!\n");
        kill(getppid(),SIGALRM);
    }
}

// end main
```

Select one:

- ☐ a. Ding
- ☐ b. MARHABA, I am awake
- ☐ c. HELLO, I am awake
- ☒ d. None of the above
- ☐ e. MARHABA, I am awake

Question 18

Not yet
answered

Marked out of
5.00

Flag
question

To terminate the process having pid=1256 we can use:

Select one:

- ☐ a. Kill(1256,SIGCONT)
- ☒ b. Kill(1256,SIGKILL).
- ☐ c. Exit(1256)
- ☐ d. kill(1250,SIGKILL)

[Clear my choice](#)



PREVIOUS ACTIVITY
CENG460_Quiz_3

Question 19

Not yet
answered

Marked out of
5.00

Flag
question

The kernel ignore all signals when an interrupt is received

Select one:

☐ True

☒ False



PREVIOUS ACTIVITY
CENG460_Quiz_3

Question 20

Not yet
answered

Marked out of
5.00

Flag
question

A process that has terminated, but not yet been waited upon, is called a zombie.

Select one:

☒ True

☐ False



PREVIOUS ACTIVITY
CENG460_Quiz_3

Jump to...



Question 20

Not yet
answered

Marked out of
5.00

Flag
question

A process that has terminated, but not yet been waited upon, is called a zombie.

Select one:

☒ True

☐ False



PREVIOUS ACTIVITY
CENG460_Quiz_3

Jump to...