

#1 Review: Consumer-Producer practice question

Consumer-Producer uses a fixed size ring buffer. s1 is initialized to 256 and s2 is initialized to zero. There are 50 producer threads & 50 consumer threads.

- Can it deadlock, if so, under what conditions?
- Is underflow possible? (underflow=Able to read/write before the start e.g. dequeue succeeds even though the data structure is empty)
- Is overflow possible? (overflow=Able to read/write after the end e.g. enqueue succeeds even though data structure is full)

Consider the following attempt. Assume buffer has 256 entries.

enqueue(value)	dequeue()
mutex_lock(m)	sem_wait(s2)
sem_wait(s1)	sem_post(s1)
sem_post(s2)	mutex_lock(m)
buffer[(in++) & 255] = value	result=buffer[(out++) & 255]
mutex_unlock(m)	mutex_unlock(m)
	return result

#2 Review: pthread practice question. What can the following code print? Assume puts is atomic.

```
void* funcA(void* ptr) { pthread_exit(((char*)ptr) + 1); }
void* funcB(void* ptr) { puts(ptr); }
```

```
int main() {
  pthread_create(&tidA,NULL,funcA,"ABC");
  pthread_create(&tidB,NULL,funcB,"XYZ");
  pthread_join(tidA, &result);
  puts(result);
  // pthread_exit(NULL)
}
```

#3 Would your answer change if main also called pthread_exit(NULL) ?

#4 Working with errors: errno, strerror,perror

What is `errno` and when is it set?

What about multiple threads?

#5 Working with `errno` and `strerror`

When is `errno` set to zero?

What are the gotchas of using `errno`?

How can you print out the string message associated with a particular error number?

What are the gotchas of using `strerror`?

#6 Interrupted system calls. AKA Correctly Handling EINTR

What is EINTR? What does it mean for `sem_wait`? `read`? `write`? `sleep`?

#7 Restarting interrupted sleep calls

e.g. SIGCHLD interrupted the sleeping parent!

```
01  ssize_t sleep_restart(int seconds) {  
02      //unsigned int remain = sleep(seconds)  
03  }
```

```
04
```

8. Correctly using `write` (IMPORTANT FOR NETWORKING)

i) May not send all bytes for slow devices (=network)

ii) May return -1 and `errno` is `EINTR`

```
01  ssize_t write_all(int fd, void*buffer, size_t len) {  
02      //Can't just call write(fd, buffer,len);  
03  }
```

```
04
```

9. Network concepts

What is IP4?

What is 127.0.0.1?

What is a port?

Can my programs listen on any port?

What is UDP? When is it used?

What is TCP? When is it used?