I'm working on my queue list, and I almost have it compiling, but I'm running into this error when making a new list:

```c
error: expected expression before 'void'
    queue *item = (queue *) newSLL(void(*d)(FILE *fp, void *));

As far as I can tell, these parameters are exactly like the parameters in the newSLL function in sll.c. I've tried editing the parameters, but nothing within reason changes the error. What am I missing?
```

Subject: Re: expected expression before 'void'
Posted by bmbaker1 on Thu, 26 Jan 2017 20:58:53 GMT

If you are making a new list... you don't need the parameters there... just the name of the display function that you want to use.

EX:
```
queue *items = newSLL(displayInteger);
```

Subject: Re: expected expression before 'void'
Posted by cewrobel on Thu, 26 Jan 2017 21:15:07 GMT

bmbaker1 wrote on Thu, 26 January 2017 14:58
If you are making a new list... you don't need the parameters there... just the name of the display function that you want to use.

EX:
```
queue *items = newSLL(displayInteger);
```

Though that's not exactly what you want to do. You shouldn't be using newSLL to initialize a queue. Just like in newSLL and in newDLL, you would use malloc to initialize a new queue. Don't forget to initialize the SLL you're using for store in the queue.
^ Yes. You should be using newQueue to construct a queue object.

I need a bigger Krieger.

---

So I would create a struct for queue, and have the call to newSLL in that:

```c
struct
newSLL(displayInteger);
end struct;
```

---

Typically a queue struct will only have one thing in it, and that’s the variable you use to store the queue (a SLL in this case). You don’t initialize anything until the newQueue function you write in queue.c

---

In your newQueue function you will use newSLL.

---

Create a struct for queue, then have a sll object in the struct. When you allocate the queue in newQueue(), that’s where you actually create the empty sll with
newSLL()
The display function is passed into newQueue() and then passed into newSLL() in the body of newSLL().