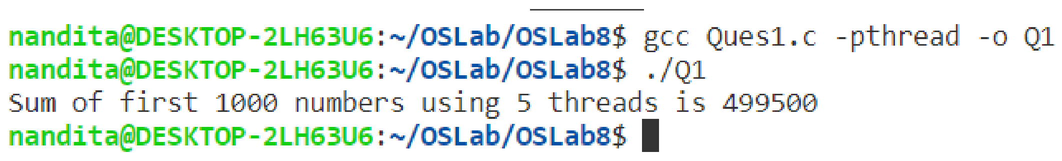
Nandita M

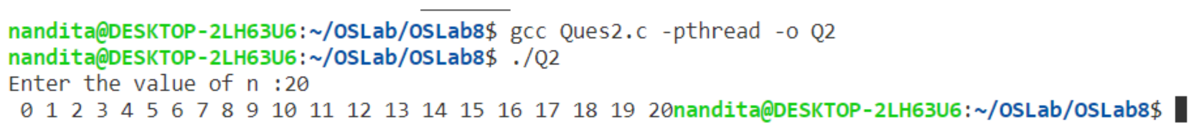
CED19I056

Ques1



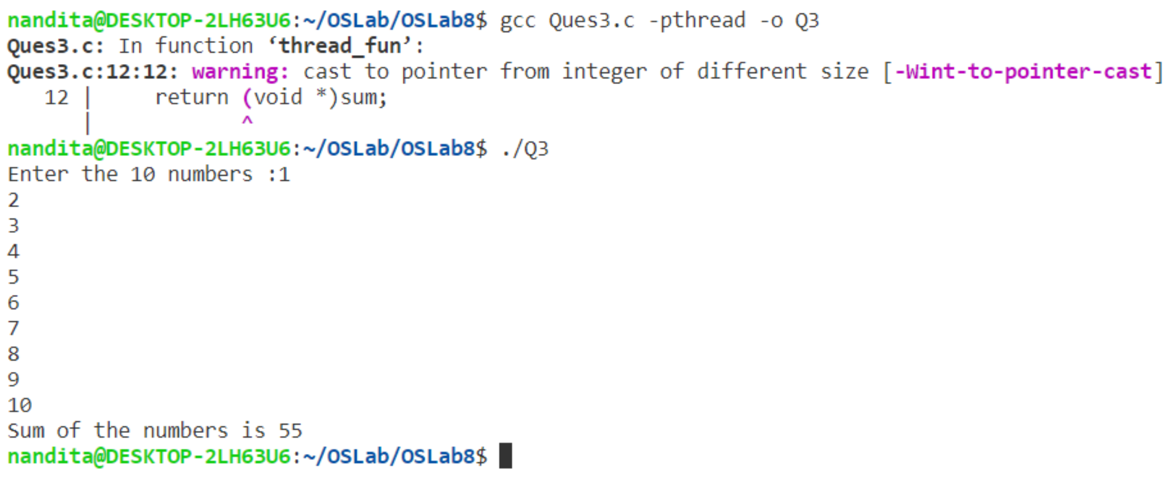
```
nandita@DESKTOP-2LH63U6:~/OSLab/OSLab8$ gcc Ques1.c -pthread -o Q1
nandita@DESKTOP-2LH63U6:~/OSLab/OSLab8$ ./Q1
Sum of first 1000 numbers using 5 threads is 499500
nandita@DESKTOP-2LH63U6:~/OSLab/OSLab8$
```

Ques2



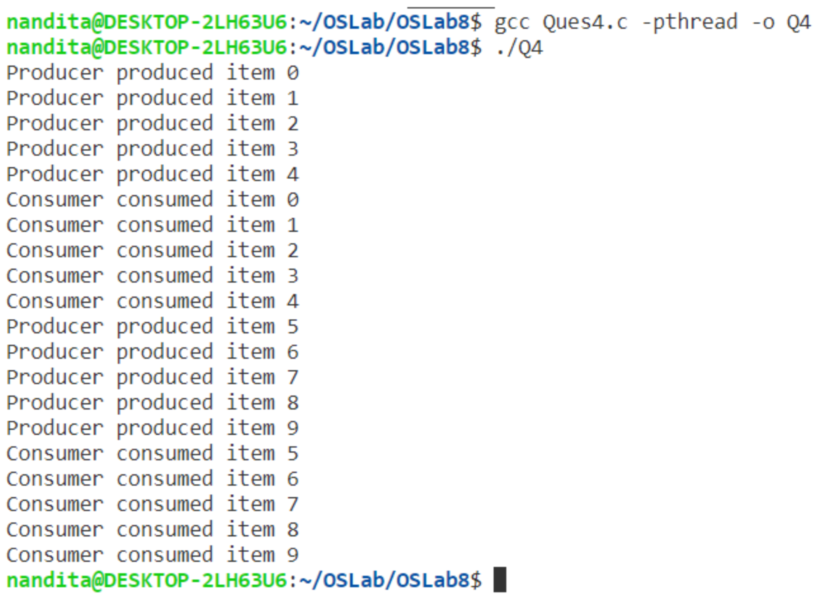
```
nandita@DESKTOP-2LH63U6:~/OSLab/OSLab8$ gcc Ques2.c -pthread -o Q2
nandita@DESKTOP-2LH63U6:~/OSLab/OSLab8$ ./Q2
Enter the value of n :20
 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20nandita@DESKTOP-2LH63U6:~/OSLab/OSLab8$
```

Ques3

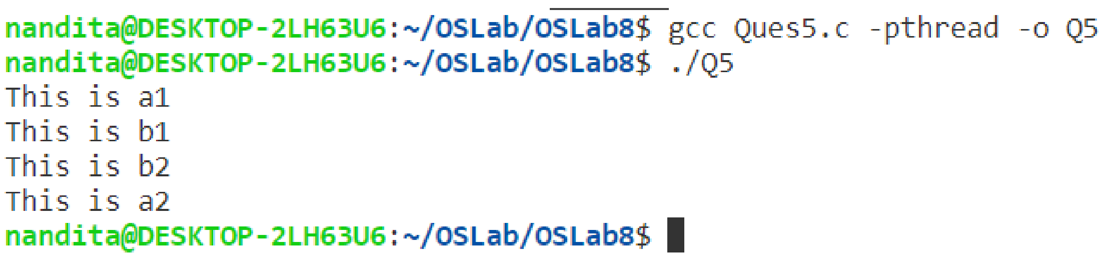


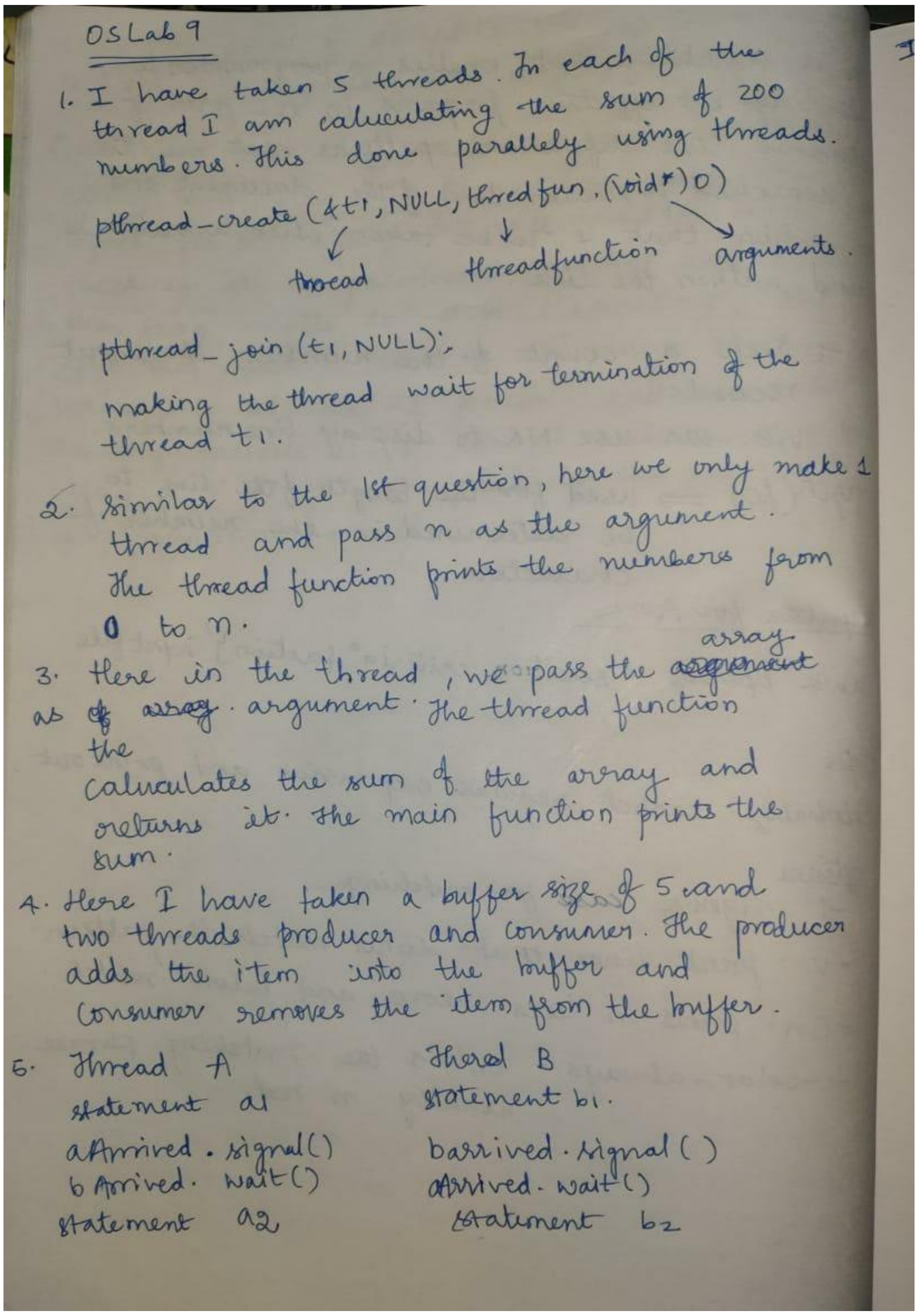
```
nandita@DESKTOP-2LH63U6:~/OSLab/OSLab8$ gcc Ques3.c -pthread -o Q3
Ques3.c: In function ‘thread_fun’:
Ques3.c:12:12: warning: cast to pointer from integer of different size [-Wint-to-pointer-cast]
   12 |     return (void *)sum;
      |            ^
nandita@DESKTOP-2LH63U6:~/OSLab/OSLab8$ ./Q3
Enter the 10 numbers :1
2
3
4
5
6
7
8
9
10
Sum of the numbers is 55
nandita@DESKTOP-2LH63U6:~/OSLab/OSLab8$
```

Ques4


```
nandita@DESKTOP-2LH63U6:~/OSLab/OSLab8$ gcc Ques4.c -pthread -o Q4
nandita@DESKTOP-2LH63U6:~/OSLab/OSLab8$ ./Q4
Producer produced item 0
Producer produced item 1
Producer produced item 2
Producer produced item 3
Producer produced item 4
Consumer consumed item 0
Consumer consumed item 1
Consumer consumed item 2
Consumer consumed item 3
Consumer consumed item 4
Producer produced item 5
Producer produced item 6
Producer produced item 7
Producer produced item 8
Producer produced item 9
Consumer consumed item 5
Consumer consumed item 6
Consumer consumed item 7
Consumer consumed item 8
Consumer consumed item 9
nandita@DESKTOP-2LH63U6:~/OSLab/OSLab8$
```

Ques5


```
nandita@DESKTOP-2LH63U6:~/OSLab/OSLab8$ gcc Ques5.c -pthread -o Q5
nandita@DESKTOP-2LH63U6:~/OSLab/OSLab8$ ./Q5
This is a1
This is b1
This is b2
This is a2
nandita@DESKTOP-2LH63U6:~/OSLab/OSLab8$
```



OSLab 9

1. I have taken 5 threads. In each of the thread I am calucculating the sum of 200 numbers. This done parallely using threads.

pthread_create (&t1, NULL, thredfun, (void*)0)

thread   threadfunction   arguments.

pthread_join (t1, NULL);

making the thread wait for termination of the thread t1.

2. Similar to the 1st question, here we only make 1 thread and pass n as the argument. The thread function prints the numbers from 0 to n.

3. Here in the thread, we pass the ~~argument~~ array as ~~of~~ ~~array~~. argument. The thread function the calucculates the sum of the array and returns it. The main function prints the sum.

4. Here I have taken a buffer size of 5 and two threads producer and consumer. The producer adds the item into the buffer and consumer removes the item from the buffer.

5. Thread A   Therad B

statement a1	statement b1.
aArrived. signal()	barrived. signal ( )
bArrived. wait()	aArrived. wait()
statement a2	statement b2