From last time

Explain briefly how a deadlock may occur (2 marks)
set of processes, each holding some resources & waiting for resources held by another process in the set

```
line   Thread A        Thread B        shared variables:
 1.    do{             do{               x, y, s
 2.      V(S1)           P(S1)
 3.      P(S2)           V(S2)         initial values:
 4.      x=x+y           P(S1)           S1=S2=0
 5.      V(S1)           y=x-y           x=y=1
 6.      V(S1)           s=s+1           s=0
 7.      P(S2)           P(S1)
 8.      print s,y       V(S2)
 9.    }while(s<7)     }while(s<7)
```

Will A ever terminate? Justify your answer. (1 mark)
sequence = e.g. A2 B2-3 A3-5 B4-6 A6 B7-8 A7-8
x=1 y=1 s=0; x=2 y=1 s=1; x=3 y=2 s=2; x=5 y=3 s=3; ...
(fibonacci) so s eventually reaches 7

ctd.

Explain the purpose of the semaphores in:

– lines 2 & 3 of both threads (1 mark)
barrier (B waits for A then A waits for B): synchronises loops

– line 5 of A & line 4 of B. (1 mark)
x=x+y happens before y=y-y, s++

– lines 6 & 7 of A & lines 7 & 8 of B. (1 mark)
another barrier: y=y-y, s++ happens before print

What is output by the print statement in line 8 of A? (3 marks)
1,1 2,2 3,3 4,5 5,8 6,13 7,21

Question

What output would you expect from Example 1?

```
5
4
exiting main thread
3
2
1
exiting child thread
```

Question: Possible Output?

Typical (just last digit of each number output):

```
Sent  0 1 2 3 4 5 6 7 8 9 0 1 2 3 4
Got  0 1 2 3 4 5 6 7 8 9 0 1 2 3 4
Sent  5 6 7 8 9 0 1 2 3 4 5 6 7 8 9
Got  5 6 7 8 9 0 1 2 3 4 5 6 7 8 9
```

Possible?:

```
Sent 01234567        89012345
Got          01234567        89012345
Sent 67890123        456789
Got          67890123      456789
```

I saw this once, I don't know why:

```
Sent 012345678         9  01  23  4
Got           012345678 90  12  34
Sent  56 7 8 9  01 2 3 4 5  67 8 9
Got  5  6 7 8 90  1 2 3 4 56  7 8 9
```