Question 1: Behold the following program:

```
#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>

int main()
{
  int i;

  for (i = 0; i < 10; i++) fork();
  printf("Hi\n");
  return 0;
}
```

How many lines will this program print?

```
#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>

int main()
{
  int p1[2], p2[2];
  int in, out;
  FILE *fin, *fout;
  int i, j;

  pipe(p1);
  pipe(p2);

  if (fork() == 0) {
    in = dup(p1[0]);
    out = dup(p2[1]);
  } else {
    in = dup(p2[0]);
    out = dup(p1[1]);
  }
  close(p1[0]);
  close(p1[1]);
  close(p2[0]);
  close(p2[1]);

  fin = fdopen(in, "r");
  fout = fdopen(out, "w");

  for (i = 0; i < 10; i++) {
    fprintf(fout, "%d\n", i);
    fflush(fout);
  }

  for (i = 0; i < 10; i++) {
    if (fscanf(fin, "%d", &j) == EOF) return 0;
    printf("j: %d\n", j);
    fflush(stdout);
  }

  return 0;
}
```

Question 2: How many lines does this program print? If you think the program never exits, enter -1.

Question 3: In the program above, how many **write**() system calls does the parent process make? If you think that the answer is "it depends", enter -1.

Question 4: In the program above, How many **read**() system calls does the parent process make? If you think that the answer is "it depends", enter -1.

Question 5: Is it possible that you'll see the 'UNIX>' prompt before all of the output of this program is printed?

Answers to today's clicker questions

Question 1

$2^{10} = 1024$.

Question 2

Let's describe what's going on:

- The parent sets up two pipes.
- The parent creates a child.
- In the parent, *in* comes from one pipe and *out* goes to the other.
- Same with the child, so that the child writes to the pipe that the parent is reading, and vice versa.
- Each of them writes 10 integers to the pipe. The **fflush()** call makes sure that the **write()** calls are made. Since the OS's pipe buffers are on the order of 4K or 8K, the **write()** calls put the bytes into those buffers, and the **fflush()** calls all return.
- Each of them reads an integer from the pipe and writes it to standard output.

So, at the end of this program, each process will have written 10 lines to standard output. The answer is 20.

Question 3

When we call **fflush()**, we're forcing a **write()** call. So there are 20 **write()** calls.

Question 4

Reading is different -- the stdio library will try to read whatever's in the pipe buffer on its first read call. So, suppose that when the parent calls **fscanf()**, the child has written everything to the pipe. Then there will only be one **read()** call. Suppose instead that when the parent calls **fscanf()**, the child is only through half of the first **for()** loop. Then there will be at least two **read()** calls. So the answer is -1 -- we don't know.

Question 5

Since the parent doesn't call **wait()**, the parent can easily exit before the child is done with its second loop. In that case, you'll get the prompt and then get some output: True.
