Question 1: How many lines are printed when you run the program to the right?

Question 2: Now, suppose you uncomment the commented line. Then which of the following is true? Just type in the letter.

- *A*: **wait()** is called too few times. There will be zombies.
- *B*: **wait()** is called too few times. There will be orphans.
- *C*: **wait()** is called too few times, but the program will return.
- *D*: **wait()** is called the correct number of times. All is good.
- *E*: **wait()** is called too many times, and the program will never return.
- *F*: **wait()** is called too many times, and there will be a segmentation violation.
- *G*: **wait()** is called too many times. Some processes will return. Some won't.
- *H*: **wait()** is called too many times, but the program will return.

```
#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>
#include <sys/wait.h>

int main()
{
   int i, status;

   for (i = 0; i < 3; i++) fork();
// for (i = 0; i < 3; i++) wait(&status);
   printf("Process %d exiting.\n", getpid());
   return 0;
}
```

Questions 3-5 refer to the code on the right, and use the following answer keys. You will answer *two* of these per question:

- *A*: The parent will become a zombie.
- *B*: The parent will become an orphan.
- *C*: The parent will become neither an orphan nor a zombie.
- *D*: It is undetermined what happens to the parent.

- *E*: The child will become a zombie.
- *F*: The child will become an orphan.
- *G*: The child will become neither an orphan nor a zombie.
- *H*: It is undetermined what happens to the child.

Question 3: In the shell, I call: `./a.out 5 10`
Question 4: In the shell, I call: `./a.out 10 5`
Question 5: In the shell, I call: `./a.out 5 5`

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

int main(int argc, char **argv)
{
  int t;

  t = (fork() == 0) ? atoi(argv[1]) : atoi(argv[2]) ;
  sleep(t);
  return 0;
}
```

Question 1

This example is similar to the example in the **fork**() lecture notes. It's a fork bomb. Let's count the processes from the top of the **for** loop:

- *i = 0*: There is just the parent, which calls **fork**() once. So, two processes reach the bottom of the **for** loop with *i = 0*.
- *i = 1*: Two processes start with *i = 1* and they both call **fork**(). So, four processes reach the bottom of the **for** loop with *i = 1*.
- *i = 2*: Four processes start with *i = 2* and they both call **fork**(). So, four processes reach the bottom of the **for** loop with *i = 2*.
- *i = 3*: Eight processes start with *i = 3* and they both call **fork**(). So, eight processes reach the bottom of the **for** loop with *i = 3*.

At that point, each of the eight processes will exit the **for** loop and print their lines. So eight lines will be printed:

```
UNIX> gcc src/clicker1.c
UNIX> ./a.out | wc
       8      24     184
UNIX>
```

Question 2

It should be clear that **wait**() is called too many times. All of those processes that were created when *i=3* don't call **fork**(), so they have no children. The ones that were created when *i=2* called **fork**() once, so they only have one child each, not three. Etc.

On the flip side, if you call **wait**() with no children, then **wait**() simply returns. So the answer is:

H: **wait**() is called too many times, but the program will return.

```
UNIX> gcc src/clicker2.c
UNIX> ./a.out
Process 83010 exiting.
Process 83012 exiting.
Process 83013 exiting.
Process 83009 exiting.
Process 83014 exiting.
Process 83011 exiting.
Process 83008 exiting.
Process 83007 exiting.
UNIX>
```

Questions 3-5

So, the parent sleeps for **atoi(argv[2])** seconds before exiting, and the child sleeps for **atoi(argv[1])** seconds before exiting.

Question 3: The child dies first. So the parent is alive and not waiting. That means that the child is a zombie. The parent is neither a zombie nor an orphan. The answers are *C* and *E*.

Question 4: The parent dies first. So the child is alive, but has no parent. That means that the child is an orphan. The parent is neither a zombie nor an orphan. The answers are *C* and *F*.

Question 5: Since they sleep for roughly the same amount of time, it's undetermined whether the child or parent dies first. So we don't know if the child becomes a zombie as in Question 3 or an orphan as in Question 4. The answers are *C* and *H*.

You'll note that in all cases the parent is neither a zombie nor an orphan. That's because the shell is waiting for it!