Program 3 Test
CSI4337

Compile and test on wind.ecs.baylor.edu
Compile with: `gcc -g -Wall -ansi -pedantic -std=c99 -lpthread`
-2 points per warning up to -10 for basic warnings
Run with: `valgrind --tool=memcheck --show-reachable=yes --leak-check=full --track-fds=yes ./<executable>`
-2 points for leaked file descriptors up to -7 (ignore fd=0, 1, and 2)
-2 points for each memory leak up to -10. VERIFY submission is responsible for leak

Part I (50 points)

0 points if program fails to compile
If program fails to operate properly, deduct 50% (min) and grade by inspection
If pthread_mutex_lock is used between processes, it shouldn't work
Must use shared memory with NO pipes. If use pipes or fails to use single, shared memory location, -40 points.
Test 10! with 3 processes. Verify completely correct answer AND memory is detached and deallocated
Test 40! with 40 processes. Verify correct answer
Test 90! With 67 processes 10 times. Verify consistent answers

Take off 5 points for each. No credit for correct answers on tests below.
Test 3 arguments – Proper usage message
Test N=-1 and P=1 – Bad N
Test N=0 and P=1 – Answer = 1
Test N=5 and P=7 – Bad P

If submission includes f.c, replace with prototype and penalize 10 points.

Compile using the following f(x) in a file named powf.c

```
double f(int x) {
  return 5.3;
}
```

This should compute 5. 3^{N+1}. Use N=7 and P=4.

Compile using the following f(x) in a file named doublef.c

```
#include <stdio.h>

double f(int x) {
  static unsigned int executions = 0;
  if (executions++ > 0) {
    fprintf(stderr, "Warning:  f(x) executed %u times\n", executions);
  }
  return 0;
}
```

This should not print a warning for N=10 and P=2.

Add usleep(1) after fork() in parent and in each loop f(x) iteration in child. Add print of process ID in each loop f(x) iteration in child. Rerun for N=15 and P=7. Verify 1) correct answer and 2) alternating child PIDs.

Part II (50 points)

0 points if program fails to compile
If program fails to operate properly, deduct 50% (min) and grade by inspection
Test 10! with 3 threads. Verify completely correct answer
Test 40! with 40 threads. Verify correct answer

Take off 5 points for each. No credit for correct answers on tests below.
Test 3 arguments – Proper usage message
Test N=-1 and P=1 – Bad N
Test N=0 and P=1 – Answer = 1
Test N=5 and P=7 – Bad P

Compile using the f(x) from powf.c. This should compute $5.\ 3^{N+1}$. Use N=7 and P=4.

Compile using the f(x) from doublef.c. This should not print a warning for N=10 and P=2.

Add usleep(1) after pthread_create() in parent and in each loop f(x) iteration in child. Add print of thread ID in each loop f(x) iteration in child. Rerun for N=15 and P=7. Verify 1) correct answer and 2) alternating child thread IDs.

For both parts

- Take off points for poor coding practices
- Verify correct use of system calls, <u>including</u> test of return values. ALL PTHREAD FUNCTION RETURNS SHOULD BE CHECKED!
- Verify use of system error messages (e.g., errno, perror, strerror, exception messages) when available. Identify overuse

-2 unused include/import
-10 for poor commenting
-2 for each unjustified numeric constant
-5 bad submission
-1 for each section of commented code
-5 if violate coding convention