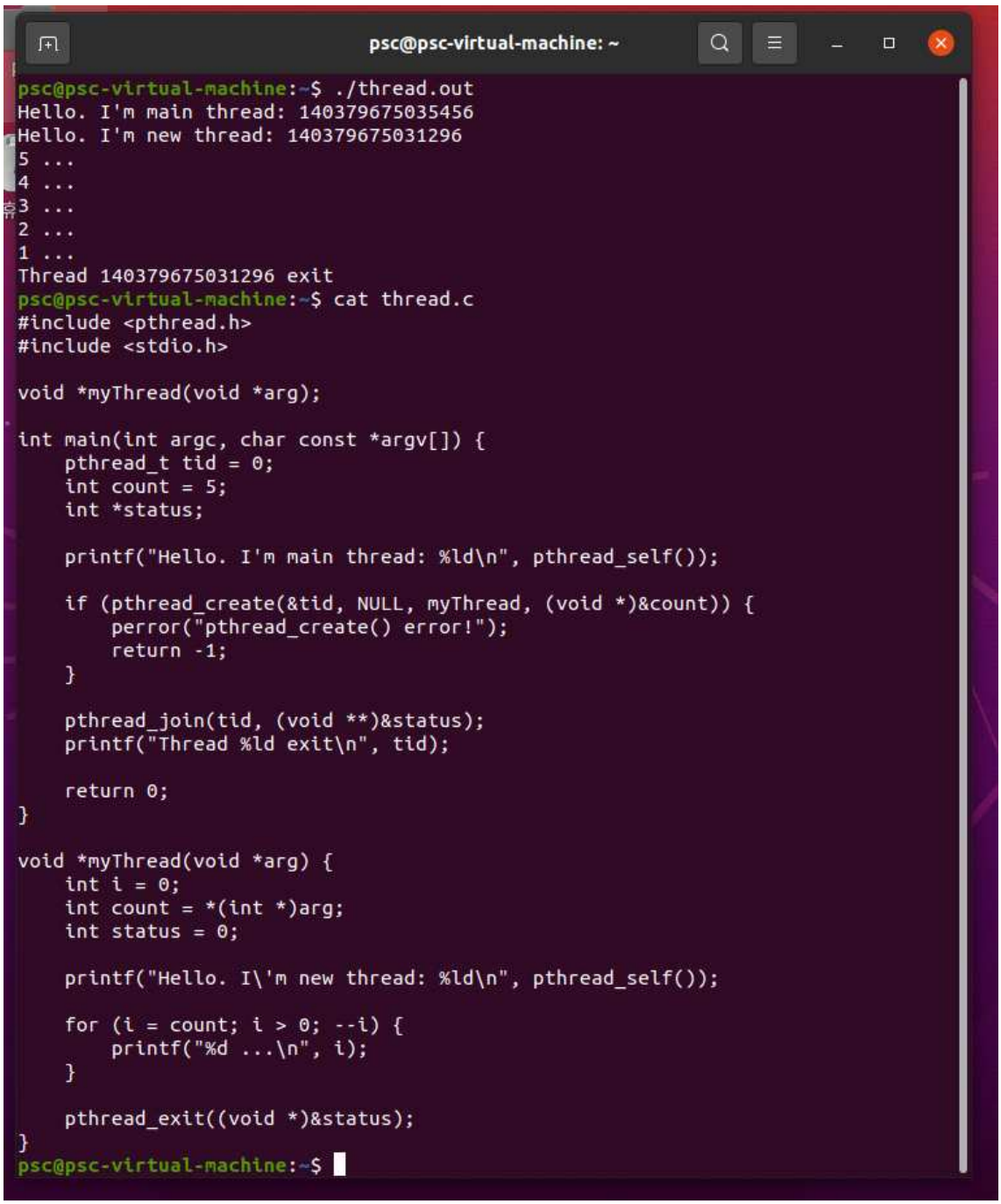
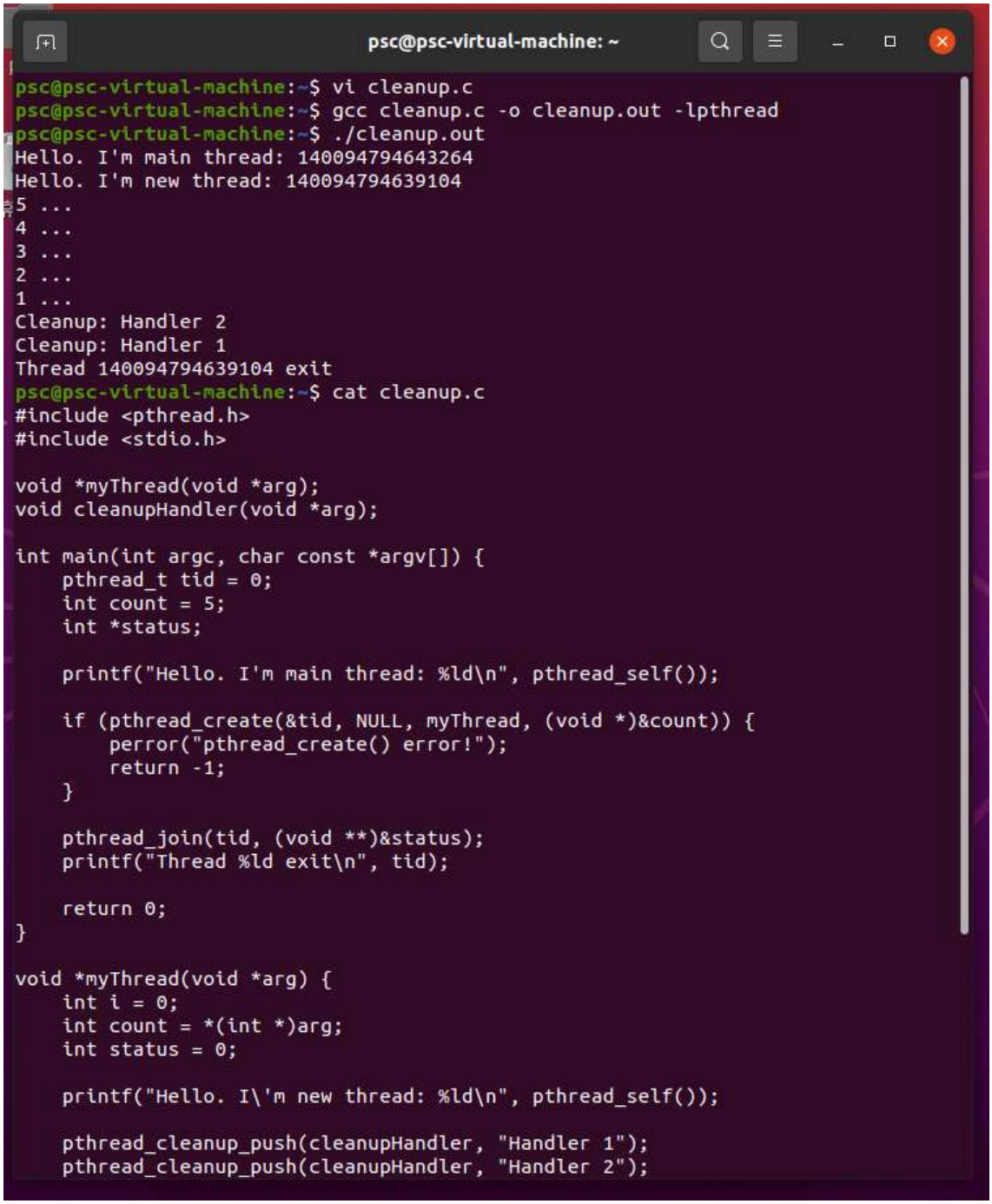
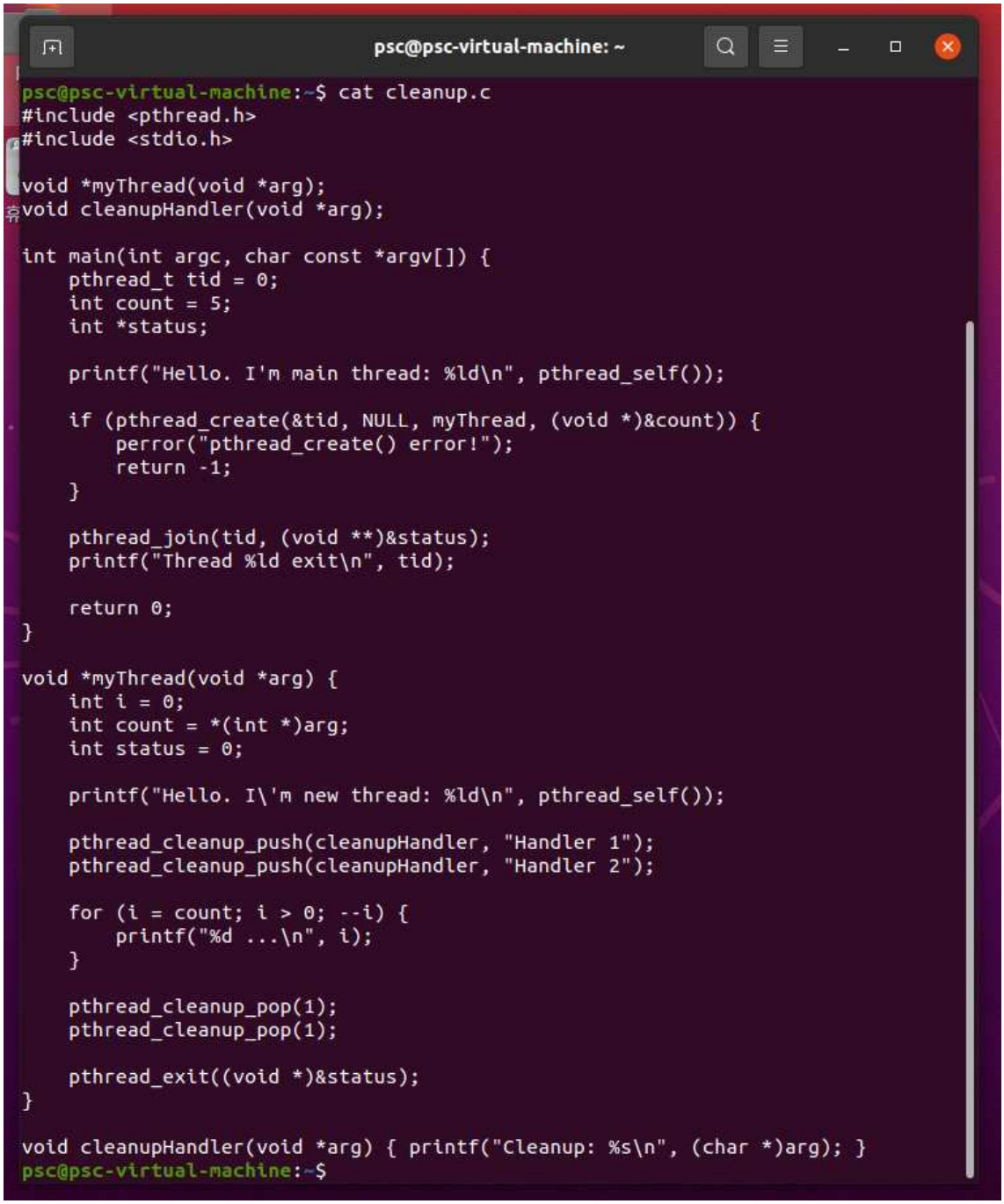
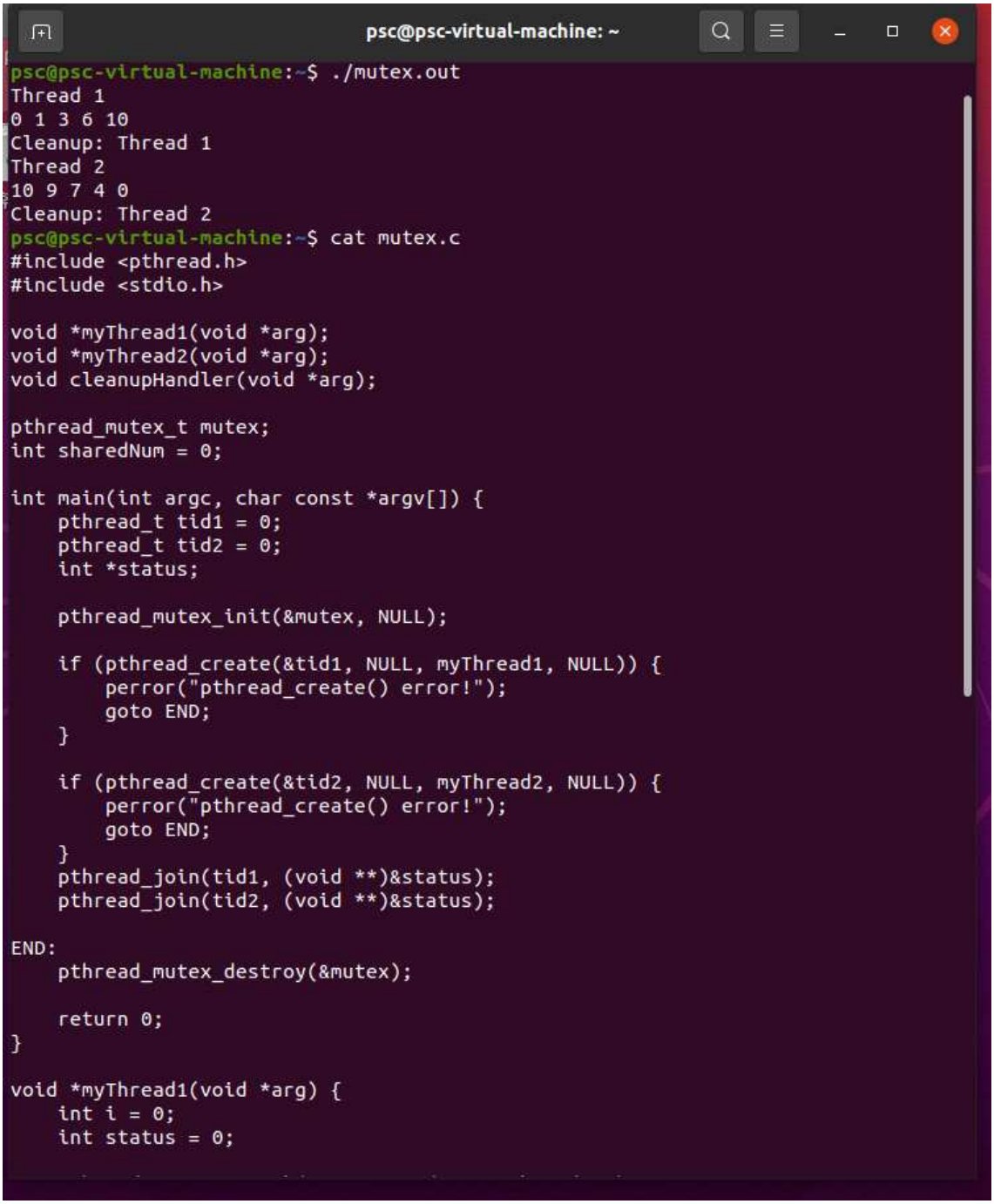
2020203002 박상천




psc@psc-virtual-machine: ~
psc@psc-virtual-machine:~$ vi cleanup.c
psc@psc-virtual-machine:~$ gcc cleanup.c -o cleanup.out -lpthread
psc@psc-virtual-machine:~$ ./cleanup.out
Hello. I'm main thread: 140094794643264
Hello. I'm new thread: 140094794639104
5 ...
4 ...
3 ...
2 ...
1 ...
Cleanup: Handler 2
Cleanup: Handler 1
Thread 140094794639104 exit
psc@psc-virtual-machine:~$ cat cleanup.c
#include <pthread.h>
#include <stdio.h>
void *myThread(void *arg);
void cleanupHandler(void *arg);
int main(int argc, char const *argv[]) {
pthread_t tid = 0;
int count = 5;
int *status;
printf("Hello. I'm main thread: %ld\n", pthread_self());
if (pthread_create(&tid, NULL, myThread, (void *)&count)) {
perror("pthread_create() error!");
return -1;
}
pthread_join(tid, (void **)&status);
printf("Thread %ld exit\n", tid);
return 0;
}
void *myThread(void *arg) {
int i = 0;
int count = *(int *)arg;
int status = 0;
printf("Hello. I\'m new thread: %ld\n", pthread_self());
pthread_cleanup_push(cleanupHandler, "Handler 1");
pthread_cleanup_push(cleanupHandler, "Handler 2");


psc@psc-virtual-machine: ~
psc@psc-virtual-machine:~$ cat cleanup.c
#include <pthread.h>
#include <stdio.h>
void *myThread(void *arg);
void cleanupHandler(void *arg);
int main(int argc, char const *argv[]) {
pthread_t tid = 0;
int count = 5;
int *status;
printf("Hello. I'm main thread: %ld\n", pthread_self());
if (pthread_create(&tid, NULL, myThread, (void *)&count)) {
perror("pthread_create() error!");
return -1;
}
pthread_join(tid, (void **)&status);
printf("Thread %ld exit\n", tid);
return 0;
}
void *myThread(void *arg) {
int i = 0;
int count = *(int *)arg;
int status = 0;
printf("Hello. I\'m new thread: %ld\n", pthread_self());
pthread_cleanup_push(cleanupHandler, "Handler 1");
pthread_cleanup_push(cleanupHandler, "Handler 2");
for (i = count; i > 0; --i) {
printf("%d ...\n", i);
}
pthread_cleanup_pop(1);
pthread_cleanup_pop(1);
pthread_exit((void *)&status);
}
void cleanupHandler(void *arg) { printf("Cleanup: %s\n", (char *)arg); }
psc@psc-virtual-machine:~$


psc@psc-virtual-machine: ~
psc@psc-virtual-machine:~$ ./mutex.out
Thread 1
0 1 3 6 10
Cleanup: Thread 1
Thread 2
10 9 7 4 0
Cleanup: Thread 2
psc@psc-virtual-machine:~$ cat mutex.c
#include <pthread.h>
#include <stdio.h>
void *myThread1(void *arg);
void *myThread2(void *arg);
void cleanupHandler(void *arg);
pthread_mutex_t mutex;
int sharedNum = 0;
int main(int argc, char const *argv[]) {
pthread_t tid1 = 0;
pthread_t tid2 = 0;
int *status;
pthread_mutex_init(&mutex, NULL);
if (pthread_create(&tid1, NULL, myThread1, NULL)) {
perror("pthread_create() error!");
goto END;
}
if (pthread_create(&tid2, NULL, myThread2, NULL)) {
perror("pthread_create() error!");
goto END;
}
pthread_join(tid1, (void **)&status);
pthread_join(tid2, (void **)&status);
END:
pthread_mutex_destroy(&mutex);
return 0;
}
void *myThread1(void *arg) {
int i = 0;
int status = 0;


psc@psc-virtual-machine: ~
END:
pthread_mutex_destroy(&mutex);
return 0;
}
void *myThread1(void *arg) {
int i = 0;
int status = 0;
pthread_cleanup_push(cleanupHandler, "Thread 1");
pthread_mutex_lock(&mutex);
puts("Thread 1");
for (i = 0; i < 5; ++i) {
sharedNum += i;
printf("%d ", sharedNum);
}
puts("");
pthread_cleanup_pop(1);
pthread_exit((void *)&status);
}
void *myThread2(void *arg) {
int i = 0;
int status = 0;
pthread_cleanup_push(cleanupHandler, "Thread 2");
pthread_mutex_lock(&mutex);
puts("Thread 2");
for (i = 0; i < 5; ++i) {
sharedNum -= i;
printf("%d ", sharedNum);
}
puts("");
pthread_cleanup_pop(1);
pthread_exit((void *)&status);
}
void cleanupHandler(void *arg) {
pthread_mutex_unlock(&mutex);
printf("Cleanup: %s\n", (char *)arg);
}
psc@psc-virtual-machine:~$