LAB 08

23K-0023

Muhammad Umar

Code Workout # 1:

a) The variable "num" is passed by reference to the "runner" function using pthread_create.
b) For now, we are using default attributes.
c)

```c
#include <stdio.h>
#include <stdlib.h>
#include <pthread.h>

void* thread_function(void* arg) {
    printf("Thread executing...\n");
    pthread_exit((void*)42);
}

int main(int argc, char *argv[]) {
    pthread_t thread;

    void* exit_status;

    pthread_create(&thread, NULL, thread_function, NULL);
    pthread_join(thread, &exit_status);

    printf("Thread exited with status: %ld\n", (long)exit_status);

    return 0;
}
```

```
umar@Muhammad-Umar:~/Desktop/Lab08$ gcc code_workout_1.c -o c1.o
umar@Muhammad-Umar:~/Desktop/Lab08$ ./c1.o
Thread executing...
Thread exited with status: 42
```

Code Workout # 2:

a)

```
umar@Muhammad-Umar:~/Desktop/Lab08$ gcc code_workout_2.c -o c2.o
umar@Muhammad-Umar:~/Desktop/Lab08$ ./c2.o
main: begin (counter = 0)
B: begin
A: begin
B: done. Counter = 14944003
A: done. Counter = 15060753
main: done with both (counter = 15060753)
```

b) Both variables are calculating different values of counter because counter is a global variable which is shared between both threads and both are updating the same counter variable.

c) Statement at line 9 is affecting the previous result because this statement is making the local instance of counter for both threads. Therefore, now instead of accessing the same global variable, both threads are updating their own local counter variable and at the end both have same values in their counter variable and global counter variable still remains zero.

```
umar@Muhammad-Umar:~/Desktop/Lab08$ gcc code_workout_2.c -o c2.o
umar@Muhammad-Umar:~/Desktop/Lab08$ ./c2.o
main: begin (counter = 0)
B: begin
A: begin
B: done. Counter = 10000000
A: done. Counter = 10000000
main: done with both (counter = 0)
```