COSC 3380 — Operating Systems

Name: Michael Osei **Due Date:** February 15 2018
Homework: Number 3

Question 4.7 Many current language specifications, such as for C and C++ are inadequate for multithreaded programs. This can have an impact on compilers and correctness of cod, as this problem illustrates. Consider the following declarations and function definition:

```
int global_positives = 0;
typedef struct list {
    struct list *next;
    double val;
}* list;

void count_positives(list l) {
    list p;
    for (p = l; p; p = p->next)
        if (p->val > 0.0)
                ++global_positives;
}
```

Now consider the case in which thread A performs

```
count_positives(<list containing only negative values>);
```

while thread B performs

```
++global_positives
```

(A) What does the function do?

(B) The C language only addresses single-threaded execution. Does the use of two parallel threads create any problems or potential problems?

Solution

(A) The function counts the number of positive elements in the linked list.

(B) From a general point of view they will be problems. But in this case there wont be any problems because the if statement in the function will not be executed for thread A and since thread B is only incrementing and global_positives is not being used anywhere else in the code there will be no problems with the program.

Question 4.8 But some existing optimizing compilers (including gcc, which tends to be relatively conservative) will "optimize" count_positives to something similar to

```
void count_positives(list l) {
    list p;
    register int r;
    r = global_positives;
    for (p = l; p; p =p->next)
        if (p -> val > 0.0) ++r;
    global_positives = r;
}
```

what problem or potential problem occurs with this compiled version of the program if threads A and B are executed concurrently?

Solution Assuming thread A is still calling count_positives with a list of only negative values then there should be no problem for thread A since it never uses r. Thread B's final value after incrementing global_positives will still be the same due to the fact that r is not modified at all global_positives