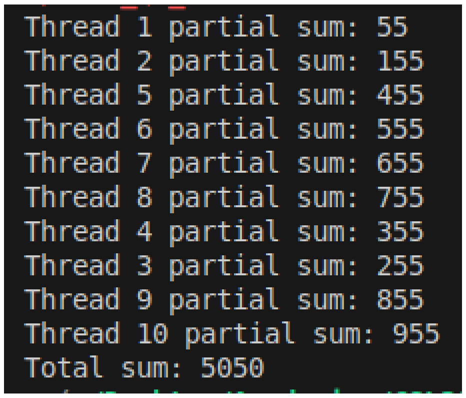
CSL301
Home Assignment 4
Amay Dixit - 12340220

Question 1.

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

#define SIZE 100
#define SEGMENT_SIZE 10
#define NUM_THREADS 11

int arr[SIZE];
int partial_sums[10];

typedef struct {
    int start;
    int end;
    int index;
} Args;

void* compute_sum(void* arg) {
    Args* a = (Args*)arg;
    int sum = 0;
    for (int i = a->start; i < a->end; ++i)
        sum += arr[i];
    partial_sums[a->index] = sum;
    printf("Thread %d partial sum: %d\n", a->index + 1, sum);
    free(a);
    pthread_exit(NULL);
}

void* total_sum(void* arg) {
    int sum = 0;
    for (int i = 0; i < 10; ++i)
        sum += partial_sums[i];
    int* total = malloc(sizeof(int));
    *total = sum;
    pthread_exit(total);
}

int main() {
    pthread_t threads[NUM_THREADS];

    for (int i = 0; i < SIZE; ++i)
        arr[i] = i + 1;

    for (int i = 0; i < 10; ++i) {
        Args* a = malloc(sizeof(Args));
        a->start = i * SEGMENT_SIZE;
        a->end = (i + 1) * SEGMENT_SIZE;
        a->index = i;
        pthread_create(&threads[i], NULL, compute_sum, (void*)a);
    }

    for (int i = 0; i < 10; ++i)
        pthread_join(threads[i], NULL);

    pthread_create(&threads[10], NULL, total_sum, NULL);

    int* total;
    pthread_join(threads[10], (void**)&total);

    printf("Total sum: %d\n", *total);
    free(total);
    return 0;
}
```

Output:



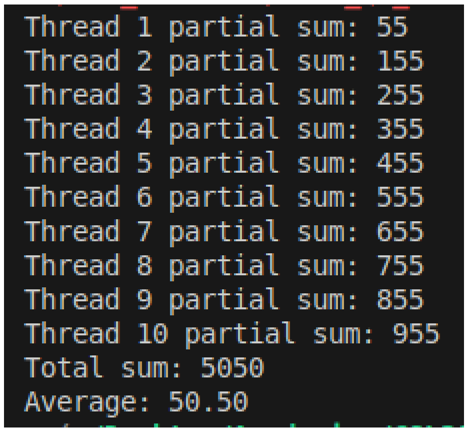
```
Thread 1 partial sum: 55
Thread 2 partial sum: 155
Thread 5 partial sum: 455
Thread 6 partial sum: 555
Thread 7 partial sum: 655
Thread 8 partial sum: 755
Thread 4 partial sum: 355
Thread 3 partial sum: 255
Thread 9 partial sum: 855
Thread 10 partial sum: 955
Total sum: 5050
```

Question 2.

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

#define SIZE 100
#define SEGMENT_SIZE 10
#define NUM_THREADS 11

int arr[SIZE];
int partial_sums[10];

typedef struct {
    int start;
    int end;
    int index;
} Args;

void* compute_sum(void* arg) {
    Args* a = (Args*)arg;
    int sum = 0;
    for (int i = a->start; i < a->end; ++i)
        sum += arr[i];
    partial_sums[a->index] = sum;
    printf("Thread %d partial sum: %d\n", a->index + 1, sum);
    free(a);
    pthread_exit(NULL);
}

void* total_sum(void* arg) {
    int sum = 0;
    for (int i = 0; i < 10; ++i)
        sum += partial_sums[i];
    int* total = malloc(sizeof(int));
    *total = sum;
    pthread_exit(total);
}

int main() {
    pthread_t threads[NUM_THREADS];

    for (int i = 0; i < SIZE; ++i)
        arr[i] = i + 1;
```

```c
    for (int i = 0; i < 10; ++i) {
        Args* a = malloc(sizeof(Args));
        a->start = i * SEGMENT_SIZE;
        a->end = (i + 1) * SEGMENT_SIZE;
        a->index = i;
        pthread_create(&threads[i], NULL, compute_sum, (void*)a);
    }

    for (int i = 0; i < 10; ++i)
        pthread_join(threads[i], NULL);

    pthread_create(&threads[10], NULL, total_sum, NULL);

    int* total;
    pthread_join(threads[10], (void**)&total);

    printf("Total sum: %d\n", *total);
    printf("Average: %.2f\n", (float)(*total) / SIZE);

    free(total);
    return 0;
}
```

Output:

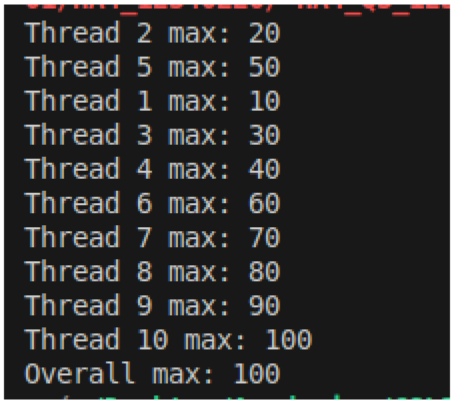


```
Thread 1 partial sum: 55
Thread 2 partial sum: 155
Thread 3 partial sum: 255
Thread 4 partial sum: 355
Thread 5 partial sum: 455
Thread 6 partial sum: 555
Thread 7 partial sum: 655
Thread 8 partial sum: 755
Thread 9 partial sum: 855
Thread 10 partial sum: 955
Total sum: 5050
Average: 50.50
```

Question 3.

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

#define SIZE 100
#define SEGMENT_SIZE 10
#define NUM_THREADS 11

int arr[SIZE];
int segment_max[10];

typedef struct {
    int start;
    int end;
    int index;
} Args;

void* find_max(void* arg) {
    Args* a = (Args*)arg;
    int max = arr[a->start];
    for (int i = a->start + 1; i < a->end; ++i)
        if (arr[i] > max)
            max = arr[i];
    segment_max[a->index] = max;
    printf("Thread %d max: %d\n", a->index + 1, max);
    free(a);
    pthread_exit(NULL);
}

void* find_overall_max(void* arg) {
    int max = segment_max[0];
    for (int i = 1; i < 10; ++i)
        if (segment_max[i] > max)
            max = segment_max[i];
    int* result = malloc(sizeof(int));
    *result = max;
    pthread_exit(result);
}

int main() {
    pthread_t threads[NUM_THREADS];
    for (int i = 0; i < SIZE; ++i)
        arr[i] = i + 1;

    for (int i = 0; i < 10; ++i) {
        Args* a = malloc(sizeof(Args));
        a->start = i * SEGMENT_SIZE;
        a->end = (i + 1) * SEGMENT_SIZE;
        a->index = i;
        pthread_create(&threads[i], NULL, find_max, (void*)a);
    }

    for (int i = 0; i < 10; ++i)
        pthread_join(threads[i], NULL);

    pthread_create(&threads[10], NULL, find_overall_max, NULL);

    int* overall_max;
    pthread_join(threads[10], (void**)&overall_max);

    printf("Overall max: %d\n", *overall_max);
    free(overall_max);
    return 0;
}
```

Output:



```
Thread 2 max: 20
Thread 5 max: 50
Thread 1 max: 10
Thread 3 max: 30
Thread 4 max: 40
Thread 6 max: 60
Thread 7 max: 70
Thread 8 max: 80
Thread 9 max: 90
Thread 10 max: 100
Overall max: 100
```