Name : Dev Adnani  
SID : 202212012

Subject : Systems Programming

Assignment : 9

1:

202212012\_Lab9\_1.c

#include <stdio.h>

#include <stdlib.h>

#include <string.h>

#include <pthread.h>

#include <unistd.h>

#include <math.h>

struct startEnd

{

int start;

int end;

};

double sroot[100];

void \*sqrRoot(void \*nos);

int main(int argc, char \*argv[])

{

pthread\_t threads[10];

int \*threadsReturn[10];

int err;

printf("Main thread id %u is starting \n", (unsigned int)pthread\_self());

int start = 0, end = 9;

struct startEnd stE;

for (int i = 0; i < 10; i++)

{

if (i != 0)

{

start = end + 1;

end = start + 9;

}

stE.start = start;

stE.end = end;

err = pthread\_create(&threads[i], NULL, sqrRoot, (void \*)&stE);

if (err != 0)

printf("Can't create the thread: %s\n", strerror(err));

err = pthread\_join(threads[i], (void \*\*)&threadsReturn[i]);

if (err != 0)

printf("Can't join the thread : %s\n", strerror(err));

}

for (int i = 0; i < 100; i++)

{

printf("Square root of : %d is %.2lf\n", i, sroot[i]);

}

printf("Main thread is %u exiting\n", (unsigned int)pthread\_self());

pthread\_exit(0);

}

void \*sqrRoot(void \*nos)

{

printf("thread id %u is starting \n", (unsigned int)pthread\_self());

struct startEnd \*thread\_p = nos;

printf("Thread with Two Params: %d :: %d\n\n", thread\_p->start, thread\_p->end);

for (int i = thread\_p->start; i <= thread\_p->end; i++)

{

sroot[i] = sqrt(i);

}

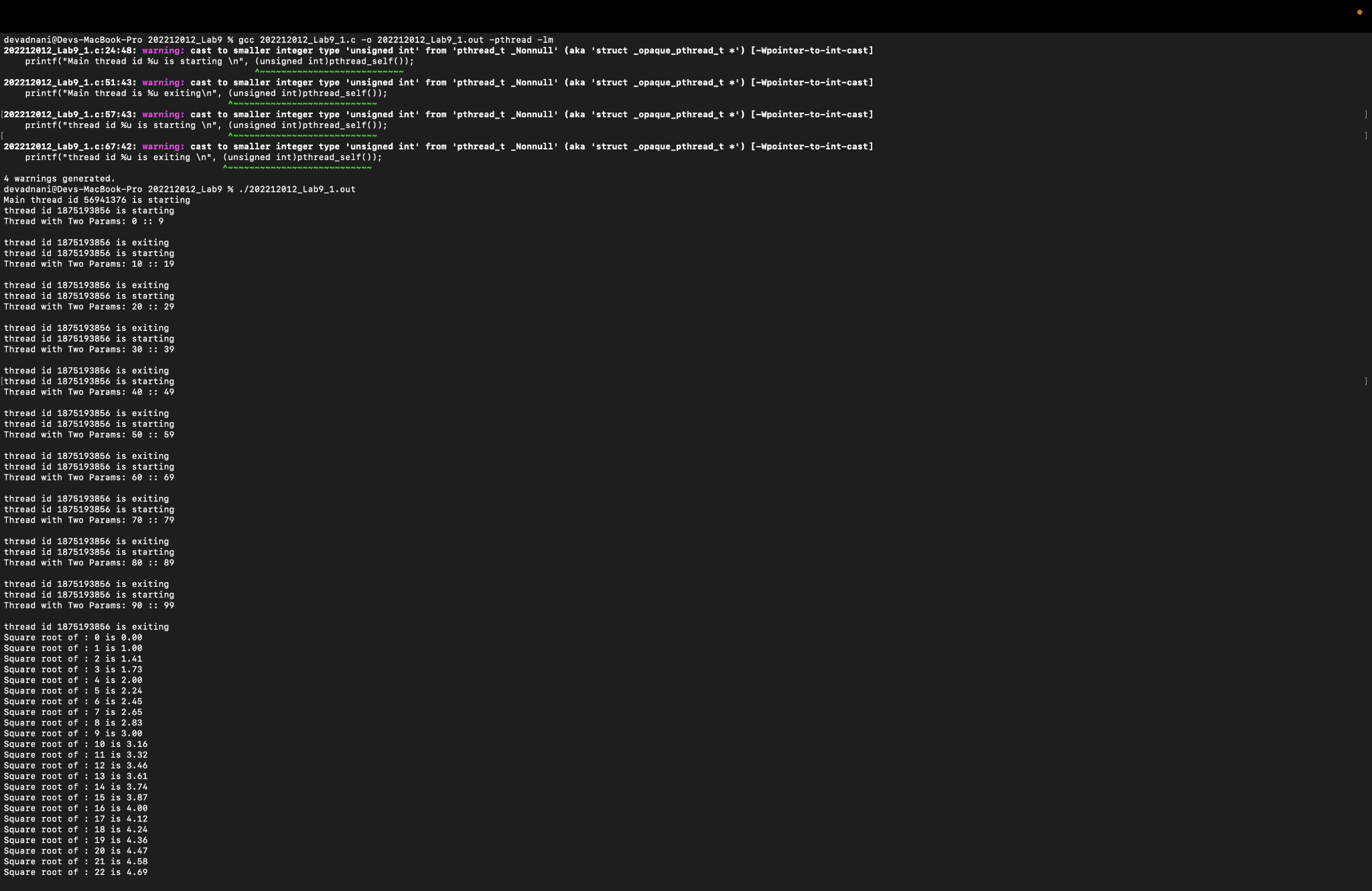
printf("thread id %u is exiting \n", (unsigned int)pthread\_self());

pthread\_exit(0);

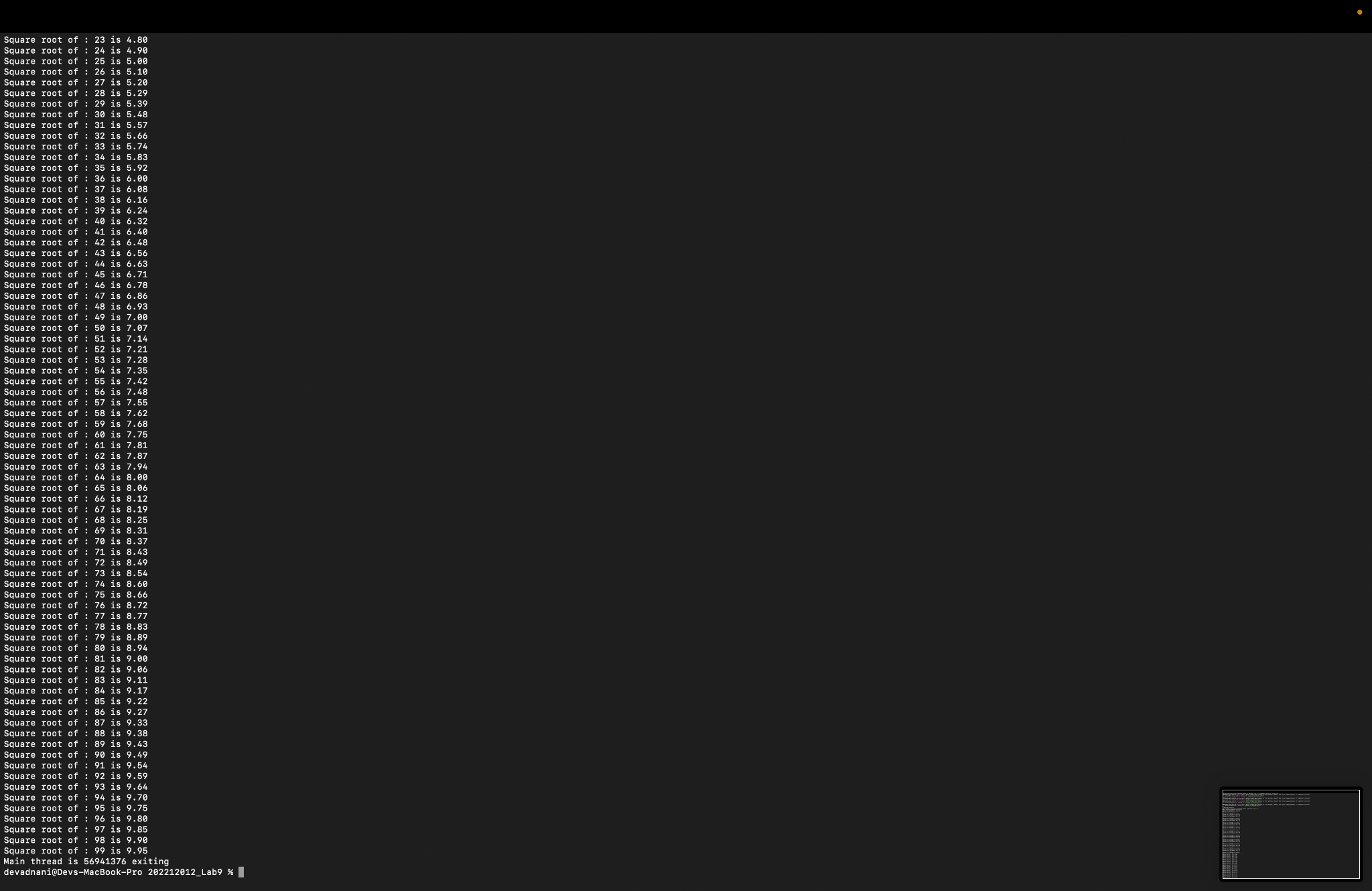
}

Screenshots :

Q1-A



Q1 - B



Q1 - C



Q2 : 202212012\_Lab9\_2\_parallel.c

#include <stdio.h>

#include <unistd.h>

#include <pthread.h>

#include <stdlib.h>

#include <sys/time.h>

#define ROW 500

#define COL 500

int mat[ROW][COL];

int cur\_row = 0;

u\_int64\_t get\_gtod\_clock\_time()

{

struct timeval tv;

if (gettimeofday(&tv, NULL) == 0)

return (u\_int64\_t)(tv.tv\_sec \* 1000000 + tv.tv\_usec);

else

return 0;

}

void \*calc\_sum(void \*arg)

{

int \*num1 = arg;

int \*local\_sum = (int \*)malloc(sizeof(int));

\*local\_sum = 0;

for (int i = 0; i < \*num1; i++)

{

for (int j = 0; j < COL; j++)

{

\*local\_sum += mat[cur\_row][j];

}

cur\_row++;

}

return ((void \*)local\_sum);

}

int main()

{

u\_int64\_t start\_time\_value, end\_time\_value, time\_diff;

int num\_threads;

long long sum = 0;

for (int i = 0; i < ROW; i++)

{

for (int j = 0; j < COL; j++)

{

mat[i][j] = i + j;

}

}

printf("Enter the no of threads to create: \n");

scanf("%d", &num\_threads);

if (num\_threads > ROW)

{

printf("Number of threads must be less than %d\n", ROW);

return 0;

}

start\_time\_value = get\_gtod\_clock\_time();

pthread\_t threads[num\_threads];

int n = ROW / num\_threads;

int rem = ROW % num\_threads;

int p;

int \*val[num\_threads];

for (int i = 0; i < num\_threads; i++)

{

p = n;

if (rem != 0)

{

p++;

rem--;

}

pthread\_create(&threads[i], NULL, calc\_sum, (void \*)&p);

pthread\_join(threads[i], (void \*\*)&val[i]);

}

for (int i = 0; i < num\_threads; i++)

{

sum += \*val[i];

}

printf("Total sum is %lld\n", sum);

end\_time\_value = get\_gtod\_clock\_time();

/\* Time difference \*/

time\_diff = end\_time\_value - start\_time\_value;

/\* display the difference \*/

printf("Time difference is %ld\n", time\_diff);

}

Screenshot :



Q2-B 202212012\_Lab9\_2\_serial.c

#include <stdio.h>

#include <unistd.h>

#include <pthread.h>

#include <stdlib.h>

#include <sys/time.h>

#define ROW 500

#define COL 500

int mat[ROW][COL];

int cur\_row = 0;

u\_int64\_t get\_gtod\_clock\_time()

{

struct timeval tv;

if (gettimeofday(&tv, NULL) == 0)

return (u\_int64\_t)(tv.tv\_sec \* 1000000 + tv.tv\_usec);

else

return 0;

}

int main()

{

u\_int64\_t start\_time\_value, end\_time\_value, time\_diff;

long long sum = 0;

for (int i = 0; i < ROW; i++)

{

for (int j = 0; j < COL; j++)

{

mat[i][j] = i + j;

}

}

start\_time\_value = get\_gtod\_clock\_time();

for (int i = 0; i < ROW; i++)

{

for (int j = 0; j < COL; j++)

{

sum += mat[i][j];

}

}

printf("Total sum is %lld\n", sum);

end\_time\_value = get\_gtod\_clock\_time();

/\* Time difference \*/

time\_diff = end\_time\_value - start\_time\_value;

/\* display the difference \*/

printf("Time difference is %ld\n", time\_diff);

}  
  
Screenshot :

