Advanced Operating Systems
Assignment # 2
Assignment deadline Sunday, 24th April 2022, 11:59 PM
Read Guidelines carefully

Question # 01

Write a program that input a character from user and print it the required times using thread.

Question # 02

Write a program that create an array of 7 threads using for loop and return Thread ID and Process ID from each thread and comments on the IDs of threads and process.

Question # 03

a) Compile the following program and observe and explain the output.

```
1
2 #include <stdio.h>
3 #include <pthread.h>
4 #include <stdlib.h>
5
6 void * thread1()
7 {
8   int c = 0;
9     while(c++ < 100)
10      printf("Hello!!\n");
11 }
12
13 void * thread2()
14 {
15   int c = 0;
16     while(c++ < 100)
17      printf("How are you?\n");
18 }
19
20 int main()
21 {
22        int status;
23        pthread_t tid1,tid2;
24
25        pthread_create(&tid1,NULL,thread1,NULL);
26        pthread_create(&tid2,NULL,thread2,NULL);
27        pthread_join(tid1,NULL);
28        pthread_join(tid2,NULL);
29        return 0;
30 }
```

b) Modify the program to create four threads using the same two functions (**thread1** and **thread2**)
c) Run both versions and include screenshots of the output.

Question # 04

Parallelize following piece of code using OpenMP with SIZE=1000. Report sequential execution time, parallel execution time (when using 2 threads, 3 threads and 4 threads), and the achieved speedup (when used 2 threads, 3 threads, and 4 threads).

Note: Your parallel formulation should provide same answer as provided by the sequential code. You will have to identify private variables, shared variables, critical section[s], and need for reduction to avoid semantic inconsistencies.

```
int key=111;

for(i=0;i<SIZE;i++){
     for(j=0;j<SIZE;j++){
          for(k=0;k<SIZE;k++){
               x= (i*i*1000/35)%1000;
               y= (j*j*1000/36)%1000;
                    z= (k*k*1000/37)%1000;
                    if(key==(x+y+z)){
                         win=win+1;
                    }

          }

     }
}
printf("total wins=%ld\n",win);
```

Question # 05

Implement a simple parallel linear search algorithm using OpenMP. The program works on an integer array of size ‘N’ filled with random numbers and a key-value to be searched. Write the parallel program to divide the search space among the multiple threads. Your program should output total number of times the occurrence of the key found in the array. [HINT: Use reduction to sum-up local counts].

Guidelines

- A single violation of guideline will lead to Zero mark in your assignment.
- You will have maximum marks if you have done the entire task.
- Only ".doc (or) .docx" file plus program codes (zip folder) should be uploaded on **Google Classroom**, Assignment would not be accepted via email, Facebook or USB flash drive etc.
- All codes should be submitted in **zip folder** and all code file names obey the following format. "RollNo_Assignment-No_Question-No.doc"
- Paste all the required outputs in the single .doc file.
- Deadlines should be kept in mind no extension in assignment dates
- This is an individual assignment. PLAGARISM IS NOT ACCEPTABLE!
- Follow the instructions as it is, otherwise your assignment would not be accepted at all.