CSE 513 Parallel Runtimes for Modern Processors, Quiz 01 Monsoon 2024
Time allocated: 12:05pm – 12:20pm

Name	
Roll Number	

Instructions:

- This is a closed book and closed notes quiz. Please be aware of strict plagiarism policy.
- For questions requiring justification, please be as concise as possible. 2-3 sentences would be the ideal size of a justification. No extra pages will be provided.

Question 1: (3 mark)
Arrange in ascending order as per the increasing runtime management cost of these entities: (One line justification for each entity)

Processes, ULTs, Tasks, and Threads

Answer:

Tasks → ULTs → Threads → Processes [+1 marks if exact order]

Task is only a combination of function pointer and arguments to this function. [+0.5 marks]

ULTs have all the states associated with a thread, but are managed inside the user space. [+0.5 marks]

Threads are managed inside the kernel space but they shares parent process's address space. [+0.5 marks]

Processes do not share address space with each other, and are managed by the kernel. [+0.5 marks]

Question 2: (1 marks)
Briefly justify if concurrency is the same as parallelism.
Answer:

Concurrency is not same as parallelism. [+0.2 marks]

Concurrency is ability to divide a program execution into small chunks where these chunks supports interleaved execution. [+0.4 marks]

Parallelism is when the execution of above chunks are actually interleaved, i.e., being executed in parallel. [+0.4 marks]

Question 3: (2 marks)
Give two reasons: Why the cost of context switches performed by the OS is **NOT** insignificant with large number of running processes/threads.
Answer:

a) Context switch requires switching the execution from user space (user stack) into kernel space (kernel stack), and it happens frequently with increasing number of threads/processes. [+1 marks]

b) Frequent invocation of kernel scheduler to decide which thread/process to run next. [+1 marks]

Question 4: (1 marks)
Which is more scalable for launching 4 tasks on a 4-core processor: work-sharing or work-stealing? (brief justification)
Answer:

Both will behave the same as creating four tasks will not have any observable overheads unlike creating several tasks. [+1 marks if correct justification]

Question 5: (3 marks)
What will be the output of the following program? (No justification required).

```
int main() {
    boost::fibers::fiber F1 ([=]() {
        cout<< “Stage-1\n”;
        boost::this_fiber::yield();
        boost::fibers::fiber F2 ([&]() {
            cout<< “Stage-2\n”;
        });
        boost::this_fiber::yield();
        cout<< “Stage-3\n”;
        F2.join();
    });
    boost::fibers::fiber F3([=]() {
        cout<< “Stage-4\n”;
    });
    F3.join();
    F1.join();
    cout<< “Stage-5\n”;
}
```

Answer:

Stage-1	[+0.2 marks]
Stage-4	[+0.6 marks]
Stage-2	[+1 marks]
Stage-3	[+1 marks]
Stage-5	[+0.2 marks]