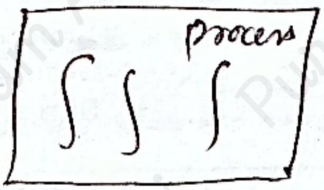
PHYSICS PRACTICAL SHEETS

Date: ..... CAMPUS
Class: ..... Experiment No.: .....
Roll No.: ..... chapter 4 Group: .....
Shift: ..... Sub.: .....
Object of the Experiment (Block Letter) Set: .....

VARIETY PRODUCT

① What is thread programming? How can you implement thread.

A thread is the smallest sequence of programmed instructions that can be managed independently by a schedular. It is always been carried out within the realm of a single process. Thread within a single process however cannot communicate with thread in other processes by sharing memory and must resort to using other forms of intercommunication such as named pipes and must resort to ~~using~~ together and sockets. As an applications becomes increasingly complex there is greater demand for computational power than can be delivered by single multicore machines. Multiple operations can be performed by modern apps at the same time. To express intra-process concurrency, developer structure program interms of threads. Threads can be used in both implicit and explicit ways. The usage of threads within a program by application developers who employ this abstraction to add parallelism is characterized as explicit threading. When an underlying APIs employs internal thread to accomplish specific task supporting application execution such as GUI rendering or garbage collection in case of virtual machine based language is refferd as implicit threading.



Three threads in execution

Within a process, a thread specifies a single controlflow which is a logical series of instruction.

creating Thread by extending Thread class

```
class Threadclass extends Thread{
    public void run(){
        System.out.println("Thread created");
    }
public static void main(String args[]){
        Threadclass t1 = new Threadclass();
            t1.start();
    }
```

using Runnable interface

```
class Threadclass R implements Runnable{
    public void run(){
    System.out.println("Thread created");
    }
public static void main(String args[]){
        Threadclass R m1 = new Threadclass R();
            Thread t1 = new Thread(m1)
                t1.start();
    }
}
```

2. What is Task programming? Write its advantages.

Task programming model is a high level multithreaded programming model. It is designed to allow maple code to be written that takes advantage of multiple processors while avoiding much of complexity of traditional multithreaded programming.

Advantages:-

- No explicit threading, users create Tasks, not threads
- Maple schedules the task to processors, so that the code scales to the number of available processors.
- Multiple algorithms written using Task programming model can run at same time without significant performance impact.
- Complex problems can be solved without requiring traditional synchronization tools such as Mutexes and conditional variable.
- If such synchronization tools are not used, the function cannot be deadlocked.
- Task function are simple and model mirrors conventional function calling.

3. Explain map reduce programming:

Map reduce is triggered by map and reduce operations in functional language such as Lisp. This model abstracts computation problems through two functions: map and reduce. All problems formulated in this way can be parallelized automatically. All data processed by mapreduce are in the form of key/value pairs. The execution happens in two phases. In the first phase, a map function is invoked once for each key/value pair and it can generate output key/value pairs as intermediate results. In the second one, all the intermediate results are merged and grouped by keys. The reduce function is called once for each key with associated values and produces o/p values as final results.

A map function takes a key/value pairs as input and produces a list of key/value pairs as output. The output key and value can be different from input key and value.

map :: (key1, value1) ⇒ list (key2, value2)

A reduce function takes a key and associated value list as input and generates a list of new values as output.

reduce :: list (key2, value2) ⇒ list (value3).

4. Explain the parallel efficiency of map reduce.

σ_D data (postmap), P processor-mapper + reducers
Assume w_D is useful work needed to be done
overheads: σ_D/p intermediate data is written by each mapper the time for transmitting it to p reducers

$$= \frac{\sigma_D}{p^2} \times p = \sigma_D/p$$

$$Emr = \frac{w_D}{p\left(\frac{w_D}{p} + \frac{2c\sigma_D}{p}\right)} = \frac{1}{1 + \frac{2c}{w}\sigma}$$

Scalable: Efficiency approaches, as useful work per data-item w grows independent of p.

If n document, m words, occuring f times per document on average so $D = nmf$

The map phase produces mp partial counts

$$\sigma = \frac{mp}{nmf} = \frac{p}{nf}$$

and $$Emr = \frac{1}{1 + \frac{2cp}{wnf}} = \frac{1}{1 + \frac{2p}{nf}}$$

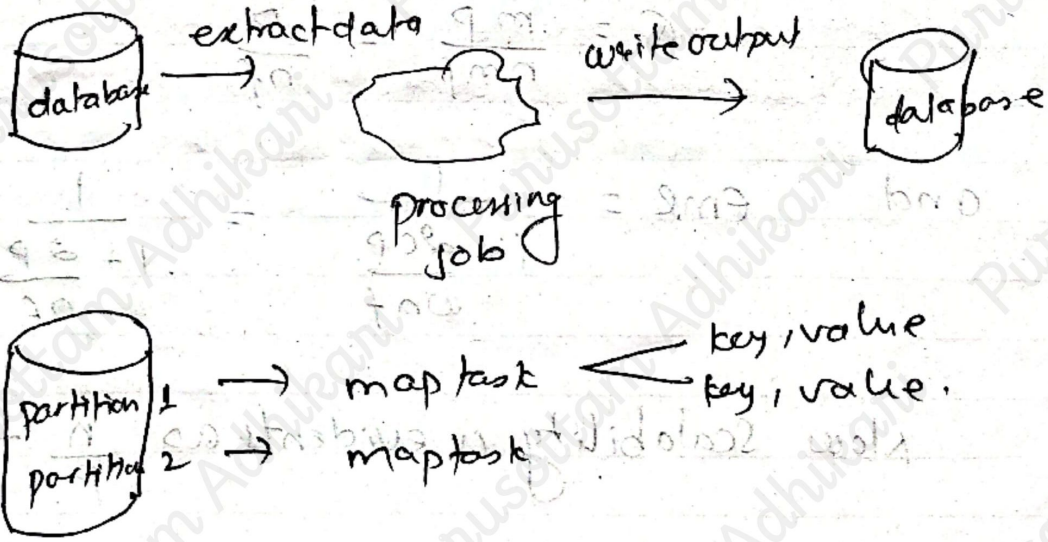
Now scalability is evidents as $\frac{n}{p} \to \infty$

5. Explain the Enterprise batch processing in mapreduce.

Batch processing is an automated job that does some computation usually done as a periodical job. It runs the processing code on a set of input called a batch. Usually the job will read the batch data from a database and store the result in same or different database.

An example of a batch processing job would be reading all the scale logs from an online shop for a single day and aggregate it into statistics for that day (number of users per country, the average spent amount etc) Doing this as a daily job could give insights into customer trends.

As the batch processing is done with cumulative transaction in a group so once batch processing has begun no user participation is necessary. It is suitable to done at the end of day.



6. Differences between Task and Thread

Task	Thread
(i) An instance of a computing program that is being executed	(i) A component of a process which is the smallest execution unit.
(ii) Heavy weight	(ii) Lightweight
(iii) switching requires interacting with OS	(iii) switching does not require interacting with OS
(iv) Each has its own memory space	(iv) Use the memory of the process they belong to
v) Require more resources	v) requires minimum resources
vi) Difficult to create	vi) easy to create.

7. Differences between Task and map reduce.

Task	map reduce
(i) A task is anything that we want to be completed that is higher-level abstraction on top of thread.	(i) map reduce is programming architecture for distributing processing.
(ii) A task describes program that may need input files and generate output files as a result of its execution & application are collected of task	(ii) It is a framework that allow design program that can process massive volumes of data in parallel
iii) most natural technique of dividing application among a group of nodes.	(iii) map reduce divide application on map and reduce phase.

iv) It takes lots of time to process lots of task or data

v) It stores less processed data

vi) difficult to get insights

iv) It takes less time to process lots of data.

v) It stores lots of data

vi) easy to get insights.