CS 553

CLOUD COMPUTING

PROGRAMMING ASSIGNMENT-1

SUBMITTED BY :

SACHIN KRISHNA MURTHY

CWID : A20354077

USER MANUAL

CPU Benchmark

There are 4 files in CPU folder.

- CPUGflops.java : Floating point operations
- CPUGiops.java : Integer point operations
- Cpu10minflops.java : 10 min for GFLOPS
- Cpu10miniops.java : 10 min for GIOPS

Instructions :

- Go to the path of the file.

Filename	CPUGflops	CPUGiops	Cpu10minflops	Cpu10miniops
Compilation	javac CPUGflops.java	javac CPUGiops.java	javac Cpu10minflops.java	javac Cpu10miniops.java
Running	java CPUGflops	java CPUGiops	Java Cpu10minflops	java Cpu10miniops

Sample Output :

Enter the choice from the list as shown below.

```
1.Using Single thread
2.Using two threads
3.Using four threads
Enter your choice:
1
----------------------For 1 Thread----------------------
Total Time :1.101
GFLOPS :11.702908158038149
```

```
1.Using Single thread
2.Using two threads
3.Using four threads
Enter your choice:
3
----------------------For 4 Threads----------------------
Total Time :2.38325
GIOPS :5.4065398446987345
```

Disk Benchmark

There are 3 files in this folder:

- DiskByte.java : For 1 Byte
- DiskKB.java : For 1 Kilo Byte
- DiskMB.java : For 1 Mega Byte

Instructions :

File Name	DiskByte.java	DiskKB.java	DiskMB.java
Compilation	javac DiskByte.java	javac DiskKB.java	javac DiskMB.java
Running	java DiskByte	java DiskKB	java DiskMB

Sample Output :

Enter the file name for sequential and random access and select your option. The files entered will be stored in the same path as of the file.

```
Enter the filename for Sequential Access:
abc
Enter the filename for Random Access:
def
-------OPTIONS------
1.Using Single thread
2.Using two threads
3.Using four threads
Enter your choice:
1
----------------------For 1 Thread----------------------
--------------------------------------------------------------------------
Sequential Access Write Time:4.56734E-4
Latency for Sequential Access Write Time :4.46029296875E-4 ms
Throughput for Sequential Access Write Time :2189.458196674651 MB/sec
--------------------------------------------------------------------------
Sequential Access Read Time:0.009211657
Latency for Sequential Access Read Time :0.0089957587890625 ms
Throughput for Sequential Access Read Time :108.55810197882965 MB/sec
--------------------------------------------------------------------------
Random Access Write Time:0.020818857
Latency for Random Access Write Time :0.0203309150390625 ms
Throughput for Random Access Write Time :0.04690759439867424 MB/sec
--------------------------------------------------------------------------
Random Access Read Time:0.010083044
Latency for Random Access Read Time :0.00984672265625 ms
Throughput for Random Access Read Time :0.09685195264445935 MB/sec
--------------------------------------------------------------------------
```

Network Benchmark

This folder consists of 12 files :

- **TCP**
  - ctcp.java : Client 1 byte
  - stcp.java : Server 1 byte
  - ctcpkb : client 1 kilo byte
  - stcpkb : server 1 kilo byte
  - ctcp64kb : client 64 kilobyte
  - stcp64kb : server 64 kilobyte

- **UDP**

  - cudp.java : Client 1 byte
  - sudp.java : Server 1 byte
  - cudpkb : client 1 kilo byte
  - sudpkb : server 1 kilo byte
  - cudp64kb : client 64 kilobyte
  - sudp64kb : server 64 kilobyte

Instructions :

- You need to write the client in one instance and server in another instance.
- You need to enter the **"IP Address of the server"** in each of the client program before running.

  For TCP :
  Socket socket=new Socket("52.36.11.80",port++);

  For UDP :
  InetAddress host=InetAddress getbyName("52.36.11.89");

- Compilation : javac filename.java (on both client and server side)

- Running : java filename (on server side first and then on client side)

Sample Output :

TCP

Server Side :

```
For 1 thread
For 2 threads
```

Client Side:

```
For 1 thread
MESSAGE SENT : OK
Write Time:19914626
For 2 threads
MESSAGE SENT : OK
MESSAGE SENT : OK
Write Time:1334279
```

UDP

Server Side :

```
For 1 thread
For 2 threads
Server socket created. Waiting for incoming data...
Server socket created. Waiting for incoming data...
Server socket created. Waiting for incoming data...
```

Client Side :

```
For 1 thread
MESSAGE SENT : OK
Write Time:16501952
For 2 threads
MESSAGE SENT : OK
MESSAGE SENT : OK
Write Time:1021237
```