1.THE MEDIUM OF IMPLEMENTATION IS **JAVA**.
2.THE FILES HAVE BEEN COMPILED AND EXECUTED ON **LINUX** SYSTEMS.
3.FOR WINDOWS THE BELOW COMMANDS MAY NOT WORK DIRECTLY. BUT, THE JAVA SOURCE FILES MENTIONED CAN BE COMPILED AND RUN.

All the **source files** are present in "**src**" directory
All **compiled class files** are present in "**classes**" directory

1. For running Min - Heap test (Project part 2)

EITHER
goto src directory
```
$ cd src
$ javac MinHeap.java MinHeapTest.java
$ java MinHeapTest
```
OR
goto classes directory
```
$ java MinHeapTest              // directly run the pre-compiled class
```

-> This code
   - generates 100 nodes (0 to 99) with random values between 1 and 1000.
   - inserts these nodes to a min heap
   - prints the contents of heap
   - deletes nodes one by one and prints them (creating sorted output)

2. For running Max Capacity Path analysis (Project part 3)

Do one of the below three options :

EITHER
a) run the shell script "PandeyNikhilesh_224002412.sh"

```
$ chmod 777 PandeyNikhilesh_224002412.sh
$ ./PandeyNikhilesh_224002412.sh
```

-> this will ask to enter the degree. Enter 6 for sparse graph and 1000 for dense graph.
The code will **(Project part 4)**
- generate 5 graphs of corresponding type (sparse / dense)
- randomly choose 5 source - destination nodes for each of the 5 graphs
- print the max capacity path, max capacity value and time taken for each of the three methods : Dijkstra (without heap), Dijkstra (with heap) and Kruskal.

OR
b) Directly run the class file

```
$ cd classes
$ java MaxCapPathAnalysis 6             (or 1000 for dense graph)
```

OR
c) Compile and run the java files (this method may be used for windows or on linux, if for some reason above - a and b - do not work)

goto src directory
```
$ cd src
$ javac *.java
$ java MaxCapPathAnalysis 6             (or 1000 for dense graph)
```

-> output in 'b' or 'c' will be same as in 'a' : **(Project part 4)**