Map Reduce Program skeleton:

We have three in Map Reduce program

1. MapClass

2. ReduceClass

3. Driver

We have ability to all these three things in one Java class

Approach 1:

```
Class OuterClass

{

      MapClass

      ReduceClass

      Driver

}
```

[OR]

Approach 2:

We have an alternative way with three individual classes..

```
      MapClass

      ReudceClass

      SomeClass

      {

            Driver

      }
```

Developer has to do:

```
import all corresponding java and hadoop packgaes;

public class OuterClass

{

      //Map Class signature
```

```
    public static class MapClass extends Mapper<keyin,
valuein, keyout, valueout>
    {
        public void setup(Context context)
        {
            Initialization before Map Process......
        }
        public void map(keyin key, valuein value Context
context) throws IOException, InterruptedException
        {
            Map level Business logic
        }

        public void cleanup()
        {
            To release any resources.... before closing map
process
        }
    }
```

MapReduce Framework provides

The Map(keyout,valueout) must equal to Reduce(keyin, valuein)

Shuffle Phase: here group by and sorting happens on map emitted keys

	input	output
map	**keyin, valuein**	**list(keyout, valueout)**
reduce	**keyin list(valuein....)**	**list(keyout, valueout)**

//Reduce Class signature

```
     public static class ReduceClass extends Redcuer<keyin,
valuein, keyout, valueout>
     {
          public void setup(Context context)
          {
               Initialization before Reduce Process......
          }
          public void reduce(keyin key, Iterable<valuein>
values, Context context) throws IOException,
InterruptedException
          {
               Reduce level Business logic
          }

          public void cleanup()
          {
               to release any resources.... before closing
reduce process
          }
     }

     //Driver Method
     public static void main(String args[])
     {
          Configuration conf = new Configuration();
          Job job = new Job(conf, "Name of the Job");
          job.setJarByClass(Set the OuterClass.class)
          job.setMapperClass(MapClass.class);
          job.setReducerClass(ReduceClass.class);
          job.setMapOutputKeyClass(mkeyout.class);
          job.setMapperOutputValueClass(mvalue.class);
          job.setOutputKeyClass(keyout.class);
          job.setOutputValueClass(valueout.class);
          job.setInputFormat(InputFormat.class);
          job.setOutputFormat(OutputFormat.class);
          FileInputFormat.addInputPath(new Path(input file
path));
          FileOutputFormat.setOutputPath(new Path(output path
file));
          System.exit(job.waitCompletionTime(true) ? 0 : 1);
     }
}
```

We have 100% in map and reduce phases.

In map, we have only one step i.e Filter and transform

In reduce, we have three thins

copy --->shuffle

sort -->shuffle

reduce --> aggregation

map 100% equals to reduce < 33.33%