Modifications to input files:

CLASS:
Vertex.java:
Functions added:

```
// this method gives incident edge list for preflow push
public LinkedList getIncidentEdgeList()
{
		return this.incidentEdgeList;
}

// this method adds a new edge to incident edges  list for a vertex.
public void addToIncidentEdgeList(Edge e) throws Exception
{
boolean isAddSucessFul = this.incidentEdgeList.add(e);
		if(!isAddSucessFul)
		{
			throw new Exception("Error in adding an edge");
		}
}
```

Edge.java:

1. Removed a line code in *insertEdge* method

Reason: The below code line is removed as the inserting an edge adding the incident edges to both sides.

Line removed: w.incidentEdgeList.addLast(e);

Method: public Edge insertEdge(Vertex v, Vertex w, Object data, Object name)

Old code:

```
public Edge insertEdge(Vertex v, Vertex w, Object data, Object name) {
		Edge e;
		 e = new Edge(v, w, data, name);
		 edgeList.addLast(e);
		 v.incidentEdgeList.addLast(e);
		w.incidentEdgeList.addLast(e);
		  return e;
	}
```

New code:

```
public Edge insertEdge(Vertex v, Vertex w, Object data, Object name) {
		Edge e;
		 e = new Edge(v, w, data, name);
		 edgeList.addLast(e);
		 v.incidentEdgeList.addLast(e);
	//  w.incidentEdgeList.addLast(e);
		  return e;
	}
```

2) Constructor :

Line added : this.getFirstEndpoint().getIncidentEdgeList().add(this);

Old code :

```
public Edge (Vertex v, Vertex w, Object data, Object name) {
 this.data = data;
this.name = name;
this.v1 = v;
```

```
this.v2 = w;
}
```

New Code :

```
public Edge (Vertex v, Vertex w, Object data, Object name) {
 this.data = data;
this.name = name;
this.v1 = v;
this.v2 = w;
this.getFirstEndpoint().getIncidentEdgeList().add(this);
}
```

Structure of the code:

FordFulkerson.java : This class computes the maximum flow in a network.
Scaling FordFulkerson.java: This class used to compute the max flow of a network. This implementation of algorithm exposes a method to calculate maximum flow of the input graph.
VertexEdge.java : This class contains the attributes vertex and Edge and the constructor which is used in FordFulkerson class.
PreFlowPush.java : This class computes the maximum flow in a network.
tcss543.java: This class contains the main function of the project

Name each routine and describe in 1-2 sentences what each routine does

Class : Ford Fulkerson:

Method: bfs-This method finds whether there is an augmented path in the graph. It uses Breadth First Search algorithm to find the path

Method: edgeBetweenVertices - This method returns an edge connecting the 2 vertices

Method: fordFulkerson - This method returns the maximum flow in the graph. It finds all the available augmented paths in the graph.

Class: VertexEdge

The class contains a constructor and getters.

Class : FordFulkersonScaling

Method : bfsWithScaling- Breadth First Search to find the path (s->t) with edge weights more than the min_weight provided.

Method : edgeBetweenVertices- Gets edge connecting start vertex(v) to end vertex(w) and return Edge v->w if exists else null.

Method: calculateScalingParam- Calculates scaling parameter of the input Graph based on the edge weights and returns the scaling parameter value for the graph.

Method : calculateMaxFlow - This method takes in a graph as input and returns the max flow of the input graph using FordFulkerson Scaling algorithm. This method also measures the time taken in milliseconds to compute the max flow.

Class : Preflow Push

Method : PreFlowPush- Constructor for loading the input graph in a Hash table.

Method : initPreflowPush- Initializes the heights and excess associated with vertices and Capacity and flow associated with edges. Gets a linked list of incident edges from GraphCode and forms residual graphs. This method is a void function and takes in the hash table(graph) as input.

Method: computeMaxFlow- This method does not take input and outputs the max flow using Preflow Push algorithm. This method measure the time taken in milliseconds while computing the max flow.

Output :
Fixed Degree Graphs

Fixed Graph 1: Number of Vertices Vs Average Runtime

No. of vertices	No.of outgoing edges	min capacity	max capacity	Input.txt	Max Flow	Runtime in(ms)Ford Fulkerson	Runtime(ms)-Scaling Ford Fulkerson	Runtime in(ms) PrePush
20	10	1	20	f20	101	15	3	20
40	10	1	20	f40	110	28	10	14
60	10	1	20	f60	117	24	12	17
80	10	1	20	f80	109	32	12	127
100	10	1	20	f100	99	25	19	215

Fixed Degree Graph 2: Number of outgoing edges Vs Average Runtime

No. of vertices	No.of outgoing edges	min capacity	max capacity	Input.txt	Max Flow	Runtime in(ms)Ford Fulkerson	Runtime(ms)-Scaling Ford Fulkerson	Runtime in(ms) PrePush
120	20	1	220	fe_20	3455	168	105	195
120	40	1	220	fe_40	3902	213	138	1399
120	60	1	220	fe_60	6834	311	212	2590
120	80	1	220	fe_80	8553	381	345	4578
120	100	1	220	fe_100	9988	563	443	6495

Fixed Degree Graph 3 : Maximum Capacity Vs Average Runtime

No. of vertices	No.of outgoing edges	min capacity	max capacity	Input.txt	Max Flow	Runtime in(ms)Ford Fulkerson	Runtime(ms)-Scaling Ford Fulkerson	Runtime in(ms) PrePush
120	10	1	200	fc_100	678	56	20	46
120	10	1	400		1991	44	27	427
120	10	1	600		2695	37	27	28
120	10	1	800		4111	59	26	363
120	10	1	1000		4741	41	25	384

Random graphs:

Random Graph 1 : Number of Vertices Vs Average Runtime

Number of vertices	Dense value	min capacity	max capacity	Input.txt	Max Flow	Ford Fulkerson	Scaling Ford Fulkerson	Preflow
20	80	1	20	r20	151	17	7	29
40	80	1	20	r40	290	52	40	232
60	80	1	20	r60	423	81	58	677
80	80	1	20	r80	551	150	85	1706
100	80	1	20	r100	777	199	207	1182
120	80	1	20	r120	1016	187	216	2519

Random Graph 2 : Maximum Capacity Vs Average Runtime

Different range of capacities								
Number of vertices	Dense value	min capacity	max capacity	Input.txt	Max Flow	Ford Fulkerson	Scaling Ford Fulkerson	Preflow
40	80	1	100	r101	1354	54	30	90
40	80	1	1000	r102	12853	53	37	235
40	80	1	3000	r106	37126	48	34	245
40	80	1	6000	r107	68171	46	32	271
40	80	1	10000	r103	144830	80	36	296
40	80	1	100000	r104	1402127	46	43	199
40	80	1	1000000	r105	13242013	58	33	88

Random Graph 3 : Density Vs Average Runtime

the number of vertices in the graph	enter dense value	min - the lower bound on edge capacities	max - the upper bound on edge capacities	Input.txt	Max Flow	Running time-FF	Running time-FFS	Running time-Preflow
100	30	10	100	graph8	1422	95	67	88
100	60	10	100	graph9	3041	256	201	239
100	90	10	100	graph10	4401	369	348	862
100	100	10	100	graph11	5267	437	397	1353

Bipartite Graphs outputs-

Table1- number of vertices vs run time and varying min and max capacity

nodes on source	nodes on sink	probability	min cap	max cap	input.txt	max flow FF/SFF/PPA	Runtime FF	Runtime SFF	Runtime PPA	total vertices
10	10	1	10	20	result1	145	31	0	47	20
20	10	1	20	30	result2	232	18	6	94	30
30	20	1	40	50	result3	904	92	31	486	50
70	30	1	50	60	result4	1667	140	78	4011	100
100	100	1	100	120	result5	10884	644	341	12107	200
100	150	1	5	50	result6	2368	725	468	4698	400
200	200	1	5	50	result7	5370	2528	2410	36730	400

Table 2- Increasing and varying the maximum capacity vs runtime

Nodes on source side	Nodes on sink side	probability	min capacity	max capacity	Input.txt	Max Flow	Runtime(ms)-Ford Fulkerson	Runtime(ms)-Scaling Ford Fulkerson	Runtime-Preflow
10	8	1	2	50	Bipartite1	194	16	0	31
10	8	1	2	500	Bipartite2	1774	0	0	16
10	8	1	2	1000	Bipartite3	3596	16	16	31
10	8	1	2	10000	bipartite4	3499	15	0	16

Table 3- varying the minimum capacity vs runtime
Probability of 1

Nodes on source side	Nodes on sink side	probability	min capacity	max capacity	Input.txt	Max Flow	Runtime(ms)-Ford Fulkerson	Runtime(ms)-Scaling Ford Fulkerson	Runtime-Preflow
10	8	1	50	400	bipartite5	2049	141	0	62
10	8	1	100	400	bipartite6	2131	31	16	47
10	8	1	200	400	bipartite7	2409	24	16	19
10	8	1	300	400	bipartite8	2640	0	0	31

Table4- minimum capacity vs runtime
Probability of 0.2

Nodes on source side	Nodes on sink side	probability	min capacity	max capacity	Input.txt	Max Flow	Runtime(ms)-Ford Fulkerson	Runtime(ms)-Scaling Ford Fulkerson	Runtime-Preflow
10	8	0.2	50	400	bipartite9	1679	46	27	172
10	8	0.2	100	400	bipartite10	944	0	3	16
10	8	0.2	200	400	bipartite11	2281	15	0	0
10	8	0.2	300	400	bipartite12	1927	8	0	5

Mesh Graph - Outputs

Table1 : Overall table with all the dimensions, runtimes of all the algorithms and Maxflow.

Input graphs	Dimensions	Capacity	Ford Fulkerson	Preflow Push	Scaling FF	Maxflow
mesh1.txt	40*20	30	370	1290	315	393
mesh2.txt	40*20	20	356	2503	107	276
mesh3.txt	40*20	40	368	1649	114	498
mesh4.txt	40*20	50	426	1154	104	614
mesh5.txt	40 * 10	50	216	392	195	656
mesh6.txt	40 * 30	50	697	1226	134	671
mesh7.txt	10*20	50	88	239	38	153
mesh8.txt	50 * 20	50	586	2633	119	762
mesh9.txt	100*100	60	17947	43283	2376	1948
mesh10.txt	100*100	20	5073	140915	2238	662
mesh11.txt	250*150	100	432868	35043085	27271	8070

Table 2 : Capacity Varied keeping Dimensions constant

Input Graphs	Dimensions	Capacity	Runtime: ff	Runtime: ppf	Runtime: scaling ff
mesh1.txt	40*20	20	356	2503	107
mesh2.txt	40*20	30	370	1290	315
mesh3.txt	40*20	40	368	1649	114
mesh4.txt	40*20	50	426	1154	104

Table 3 : Column dimension varied keeping Capacity and Row dimension constant

Input Graphs	Column dimn	Capacity	Runtime: ff	Runtime: ppf	Runtime: Scaling ff
mesh5.txt	40 * 10	50	216	392	195
mesh4.txt	40*20	50	426	1154	104
mesh6.txt	40 * 30	50	697	1226	134

Table 4 : Row Dimension varied keeping Capacity and Column dimension constant.

Input Graphs	Row dimn	Capacity	Runtime: ff	Runtime: ppf	Runtime: Scaling ff
mesh7.txt	10*20	50	88	239	38
mesh4.txt	40*20	50	426	1154	104
mesh8.txt	50 * 20	50	586	2633	119