Description	Math	AMPL	OR-Tools (Direct)	OR-Tools (Dataframes)
Sets	Let <name> denote the set of <something> Let <name> be a subset of $\{(i,j) : i \in$ <this>$, j \in$ <that>$\}$	set <NAME>; set <NAME> within <this> cross <that>;	<NAME> = list(<something>)	<NAME> = list(<something>.index)
Parameters	Let <name>$_i$ denote <name> for $i \in$ <SET>	param <name> {i in <SET>};	<name> = dict(<name>)	<name> = <SET>[<name>].to_dict()
Input Check	<name>$_i \geq 0$ for $i \in$ <SET>	check {i in <SET>}: <name>[i] >= 0;	for i in <SET> assert <name>[i] >= 0	
Model			m = OR.Solver(<name>, <solver>) OR.Solver.GLOP_LINEAR_PROGRAMMING: Open-source LP solver OR.Solver.CBC_MIXED_INTEGER_PROGRAMMING: Open-source MIP solver OR.Solver.GUROBI_MIXED_INTEGER_PROGRAMMING: Commercial MIP solver	
Variables	Let $x_i \geq 0$ denote <name> for $i \in$ <SET>	var x {i in <SET>} >= 0;	x = {} for i in <SET>: x[i]= m.NumVar(0, m.infinity(), ("(%s)" % (i)))	
	(Integer variables)	var x {i in <SET>} >= 0 integer;	x = {} for i in <SET>: x[i]= m.IntVar(0, m.infinity(), ("(%s)" % (i)))	
	Let $x_{ij} \geq 0$ denote <name> for $(i,j) \in$ <SET>	var x {(i,j) in <SET>} >= 0;	x = {} for i,j in <SET>: x[i,j]= m.NumVar(0, m.infinity(), ("(%s, %s)" % (i,j)))	
Summation	$\sum_{i \in \texttt{<SET>}} x_i$ $\sum_{(i,j) \in \texttt{<SET>}} x_{ij}$	sum {i in <SET>} x[i]; sum {(i,j) in <SET>} x[i,j];	sum(x[i] for i in <SET>) sum(x[i,j] for i,j in <SET>)	
Objective Function	max <expression> min <expression>	maximize <Some_Name>: <expression>; minimize <Some_Name>: <expression>;	m.Maximize(<expression>) m.Minimize(<expression>)	

Constraints	such that <expression>
	such that <expression> for $i \in$ <SET>
	
	such that <expression> for $(i,j) \in$ <SET>
		subject to <Some_Name>: <expression>;
		subject to <Some_Name> {i in <SET>}:
		  <expression>;
		subject to <Some_Name> {(i,j) in <SET>}:
		  <expression>;
			m.Add(<expression>)
			for i in <SET>:
			  m.Add(<expression>)
			for i,j in <SET>:
			  m.Add(<expression>)	

Set Data		set <NAME> := <item1> <item2> <item3>;	<name> = [<item1>,<item2>]	<SET>,<name>

Parameter Data	<name>$_i$ for $i \in$ <SET>
	
	<name>$_{ij}$ for $(i,j) \in$ <SET>
		param <name> :=
		  i <name>$_i$;
		param <name> :=
		  i j <name>$_{ij}$;
			<name> = {i:<name>$_i$,
			          j:<name>$_j$}
			<name> = {(i,j):<name>$_{ij}$,
			          (a,b):<name>$_{ab}$}
				i,<name>$_i$
				j,<name>$_j$

Min-Cost Flow	Consider the directed graph $G = (N, E)$	set NODES; set EDGES within (NODES cross NODES);	NODES = list(nodes) EDGES = list(edges)	NODES = list(nodes.index) EDGES = list(edges.index)
	Let b_i be the supply at node $i \in N$ Let c_i be the cost of edge $(i,j) \in E$ Let u_i be the upper bound of edge $(i,j) \in E$	param b {i in NODES}; param c {(i,j) in EDGES}; param u {(i,j) in EDGES};	b = dict(b) c = dict(c) u = dict(u)	b = nodes["b"].to_dict() c = edges["c"].to_dict() u = nodes["u"].to_dict()
			m = OR.Solver("mincostflow", OR.Solver.GLOP_LINEAR_PROGRAMMING)	
	Let $x_{ij} \geq 0$ be the flow on edge $(i,j) \in E$	var x {(i,j) in EDGES} >= 0;	x = {} for i,j in EDGES: x[i,j]= m.NumVar(0, m.infinity(), ("(%s, %s)" % (i,j)))	
	$\max \sum_{(i,j) \in E} c_{ij} x_{ij}$	minimize Total_Cost: sum (i,j) in EDGES c[i,j]*x[i,j];	m.Minimize(sum(c[i,j] * x[i,j] for i,j in EDGES))	

	s.t. $\quad x_{ij} \leq u_{ij} \quad \forall (i,j) \in E$	subject to Capacity {(i,j) in EDGES}:
		  x[i,j] <= u[i,j];
			for i,j in EDGES:
			  m.Add(x[i,j] <= u[i,j])	

	s.t. $\sum_{(i,j) \in E} x_{ij} - \sum_{(j,i) \in E} x_{ji} = b_i \quad \forall i \in N$	subject to Supply i in NODES:
		  (sum (i,j) in EDGES x[i,j] -
		   sum (j,i) in EDGES x[j,i] = b[i]);
			for k in NODES:
			  m.Add(sum(x[i,j] for i,j in EDGES if i == k)
			      - sum(x[i,j] for i,j in EDGES if j == k) == b[k])	