




<<Java Class>>


 **Heuristic**

Heuristic

 Heuristic()

 search(Graph):void


<<Java Class>>


 **Graph**


Heuristic


□ n: int


□ weights: double[]

 Graph(int)

 getAllNodes():List<Node>


 setAllNodes(List<Node>):void

 setWeight(int,int,double):void

 getWeights(int,int):double

-allNodes 0..*

<<Java Class>>

 **Node**


Heuristic


□ element: Object


□ visited: boolean


□ neighbours: List<Node>


□ index: int

 Node(Object,int)


 isVisited():boolean


 getIndex():int


 setIndex(int):void


 addNeighbours(Node):void


 getElement():Object

 setElement(Object):void

 isVisted():boolean

 setVisted(boolean):void

 getNeighbours():List<Node>


 setNeighbours(List<Node>):void

<<Java Class>>

 **Main**

Heuristic

 Main()

 ^Smain(String[]):void