- initialization(graph: Graph): void - resetTraversibility(path: Path): void - pheromoneEvaporation(graph: Graph, evaporationCoefficent): void Graph - vertexes: Vertex[*]{unique} edges: Edge[*]{unique} + Graph(vertexes: Vertex[*]{unique}, edges: Edge[*]{unique}) + addVertex(vertex: Vertex): bool + deleteVertex(vertexId: int): bool + addEdge(edge: Edge): bool + deleteEdge(edge: Edge): bool Edge Vertex - vertexes: Set {readOnly} - id: int {readOnly} weight: int {readOnly} - edges: Edge[*]{unique} desirability: float {readOnly} + Vertex(id: int) + depositedPheromone: float 2 + getId(): int + traversible: bool + addEdge(edge: Edge): bool + Edge(v1: Vertex, v2: Vertex, weight:int) + removeEdge(edge: Edge): bool + getWeight(): float + getTraversibleEdges(): Edge[*]{unique} + getDesirability(): float + getEdges(): Edge[*]{unique} + getVertexes(): Vertex[2]{unique} + getOtherEnd(vertex: Vertex): Vertex + delete(): void Path beginning: Vertex {readOnly} edges: Edge[*]{unique} {readOnly} cost: int {readOnly} + Path(beginning: Vertex, edges: Edge[*]{unique}, cost: int) + getBeginning(): Vertex + getEdges(): Edge[*]{unique} + getCost(): int + printPath(): void Ant + Ant()

+ createPath(start: Vertex, end: Vertex): Path

+ depositPheromone(path: Path): void

- chooseEdgeToTraverse(possibleEdges: Edge[*]{unique}, pheromoneInfluence: float, desirabilityInfluence: float): Edge

ACO_ShortestPath

+ main(pheromoneInfluence: float, desirabilityInfluence: float, evaporationCoefficent: float): void

+ findShortestPath(graph: Graph, start: Vertex, end: Vertex, iter_num: int): Path

pheromoneInfluence: floatdesirabilityInfluence: floatevaporationCoefficent: float