## 2 non-trivial methods: **Graph class** 1)findDogRelationshipByMenu(): this function takes in two dog breed names and is able to check if these dogs have an Edge connecting them, if they do, Benjamin Tan + Sepehr Mohebbi | April 19, 2022 the directed/undirected edge will be printed out with its weight, as well as both dogs' full information 2) aPathFindByMenu(): this function takes in two dog IDs and finds all paths connecting them. Once it does, it print outs all these paths. Edges Menu -V1:Vertex -V2:Vertex Graph -graph: Graph\* -weight:int 0..\* #vertexArr : vector <Vertex> +setter/getter +Menu(Graph\*) #edgeArr: vector <Edge> +userInput(string&): void + directedToString():string + undirectedToString():string +userInput(T&, string) +DisplayMenu(): void +listDatabase: void +listDatabaseWithEdges(): void +addVertexByMenu(): void +Graph() +addVerticesByMenu(): void #Graph(vector<Vertex>&, vector<Edge>&) +addEdgesByMenu(): void #ReturnVertexByName(string): Vertex +removeVertexByMenu(): void 2 # checkVertexInArr (int, vector<Vertex>&) const : bool +printDogByMenu(): void #returnVertexPtrbyID(int) const : const Vertex\* Vertex +findDogRelationshipByMenu(): void +addVertex(const Vertex&): void +pathFinderByMenu(): void +addVertices(const vector<Vector>&): void - id:int +aPathFindByMenu(): void +removeVertex(string): void - dog:DogBreeed +addEdge(const Edge&): void 0..\* - idCounter:static int +addEdges(const vector<Edge>&): void +remove(const Edge&): bool Techniques used +getDog():const Dogbread& +searchEdge(const Edge&) const : bool +printAllDogs() const : void +getId():int 1) Inheritance used to +printDog(const Vertex&) const : void +toString():string +clean(): void create the directed +queryByEdge(const Edge&) const : void graph and undirected +queryByVertexValue(string) const : void graph functionality +allPathFinder() const : void ~Graph() 2) Polymorphism used to make Graph and abstract class, with DirectedGraph and UndirectedGraph being the concrete classes with their own **▼**1 implementations of the virtual member DogBreed functions - breed:string 3) operator overloading - height:doube (<<) used to display - lifeExpectancy:double DirectedGraph undirectedGraph graph information: vertices and edges. +setter/getter +DirectedGraph() +UndirectedGraph() ex: cout << graph +DirectedGraph(vector<Vertex>&, vector<Edge>&) +UndirectedGraph(vector<Vertex>&, vector<Edge>&) +toString() const: string +toString() const: string +thePathFinder(int,int,vector<Vertex>&) const : void +thePathFinder(int,int,vector<Vertex>&) const : void +thePathFinder(int, vector <Vertex>&> const : void +thePathFinder(int, vector <Vertex>&> const : void +pathFinder (int, int) const : void +pathFinder (int, int) const : void +checktype() const : bool +checktype() const : bool