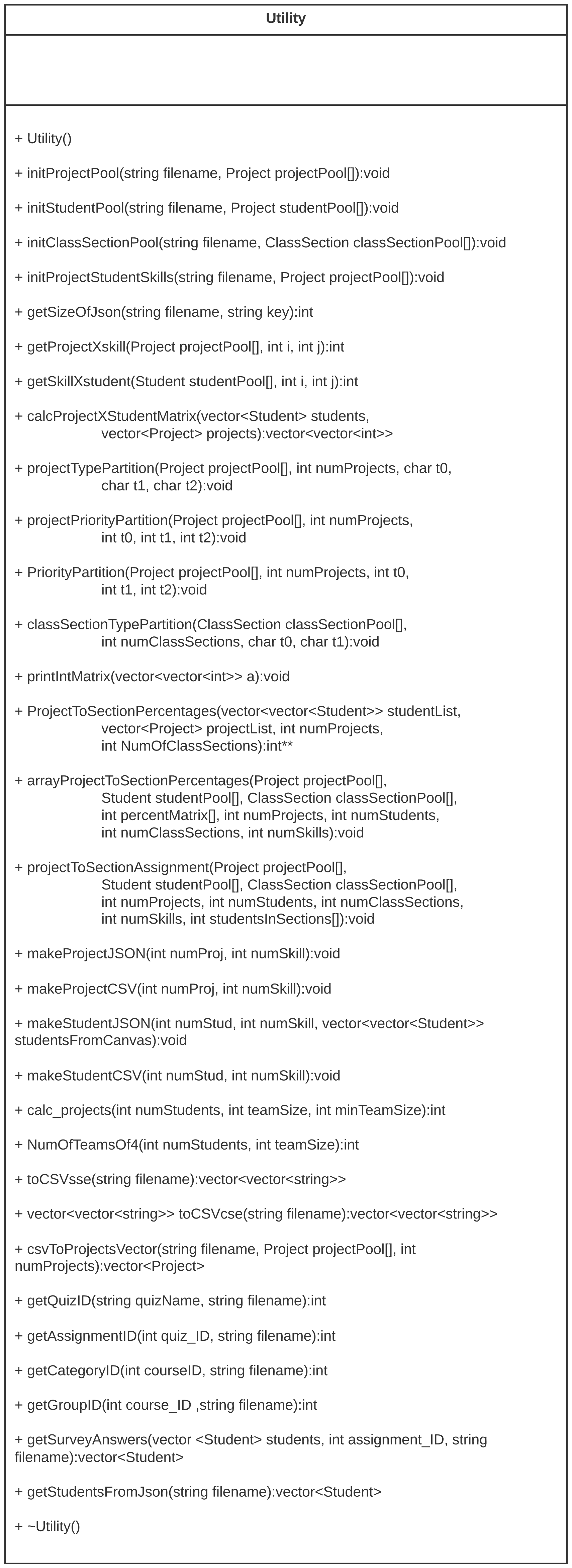

Utility

```
+ Utility()

+ initProjectPool(string filename, Project projectPool[]):void

+ initStudentPool(string filename, Project studentPool[]):void

+ initClassSectionPool(string filename, ClassSection classSectionPool[]):void

+ initProjectStudentSkills(string filename, Project projectPool[]):void

+ getSizeOfJson(string filename, string key):int

+ getProjectXskill(Project projectPool[], int i, int j):int

+ getSkillXstudent(Student studentPool[], int i, int j):int

+ calcProjectXStudentMatrix(vector<Student> students,
              vector<Project> projects):vector<vector<int>>

+ projectTypePartition(Project projectPool[], int numProjects, char t0,
              char t1, char t2):void

+ projectPriorityPartition(Project projectPool[], int numProjects,
              int t0, int t1, int t2):void

+ PriorityPartition(Project projectPool[], int numProjects, int t0,
              int t1, int t2):void

+ classSectionTypePartition(ClassSection classSectionPool[],
              int numClassSections, char t0, char t1):void

+ printIntMatrix(vector<vector<int>> a):void

+ ProjectToSectionPercentages(vector<vector<Student>> studentList,
              vector<Project> projectList, int numProjects,
              int NumOfClassSections):int**

+ arrayProjectToSectionPercentages(Project projectPool[],
              Student studentPool[], ClassSection classSectionPool[],
              int percentMatrix[], int numProjects, int numStudents,
              int numClassSections, int numSkills):void

+ projectToSectionAssignment(Project projectPool[],
              Student studentPool[], ClassSection classSectionPool[],
              int numProjects, int numStudents, int numClassSections,
              int numSkills, int studentsInSections[]):void

+ makeProjectJSON(int numProj, int numSkill):void

+ makeProjectCSV(int numProj, int numSkill):void

+ makeStudentJSON(int numStud, int numSkill, vector<vector<Student>>
studentsFromCanvas):void

+ makeStudentCSV(int numStud, int numSkill):void

+ calc_projects(int numStudents, int teamSize, int minTeamSize):int

+ NumOfTeamsOf4(int numStudents, int teamSize):int

+ toCSVsse(string filename):vector<vector<string>>

+ vector<vector<string>> toCSVcse(string filename):vector<vector<string>>

+ csvToProjectsVector(string filename, Project projectPool[], int
numProjects):vector<Project>

+ getQuizID(string quizName, string filename):int

+ getAssignmentID(int quiz_ID, string filename):int

+ getCategoryID(int courseID, string filename):int

+ getGroupID(int course_ID ,string filename):int

+ getSurveyAnswers(vector <Student> students, int assignment_ID, string
filename):vector<Student>

+ getStudentsFromJson(string filename):vector<Student>

+ ~Utility()
```

Learn about this template

UML class diagrams map out the structure of a particular system by modeling its classes, attributes, operations, and relationships between objects.

To customize this template:

- *Click on any shape and type the information you would like to include.*
- *Add and arrange class shapes as needed.*
- *Update cardinality.*
    - *Click on a line and navigate to the properties bar to adjust the endpoints.*
    - *Click on a line and hover over the gear icon to add multiplicities.*
    - *Add additional lines by hovering over a shape and clicking the red dot*

UML Class Diagram Tutorials

(Hold Shift + ⌘ or Ctrl, then click)

Watch a UML class diagram tutorial

Read about UML class diagrams

Watch Lucidchart basic tutorials