

API Documentation

students

display_menu()

add_student()

add_grade()

average()

save_to_file()

load_from_file()

delete_student()

delete_grade()

built with 

undocumented

► View Source

students: Dict[str, List[float]] = {'Max': [85, 92, 78], 'Marina': [90, 88], 'Ashley': [65, 74, 50], 'Bonnie': [99, 87]}

def display_menu() -> None:

▼ View Source

```

20 def display_menu() -> None:
21     """
22     Display the main menu options.
23     """
24     print("-" * 30)
25     print(
26         """1. Add student
27     2. Add grade
28     3. Show average for student
29     4. Show all students
30     5. Save to file
31     6. Load from file
32     7. Exit"""
33     )
34     print("-" * 30)

```

Display the main menu options.

def add_student(students: Dict[str, List[float]], name: str) -> None:

▼ View Source

```

37 def add_student(students: Dict[str, List[float]], name: str) -> None:
38     """
39     Add a new student to the students dictionary.

```

API Documentation

students

display_menu()

add_student()

add_grade()

average()

save_to_file()

load_from_file()

delete_student()

delete_grade()

built with 

```

40
41     Parameters:
42         students (dict): The dictionary of students.
43         name (str): The name of the student to add.
44
45     Returns:
46         None
47     """
48     if name not in students:
49         students[name] = []
50     else:
51         print(f"Student {name} already exists.")

```

Add a new student to the students dictionary.

Parameters: students (dict): The dictionary of students. name (str): The name of the student to add.

Returns: None

def add_grade(students: Dict[str, List[float]], name: str, grade: float) -> **None**: [▼ View Source](#)

```

54 def add_grade(students: Dict[str, List[float]], name: str, grade: float) -> None:
55     """
56     Add a grade to a student's List of grades.
57
58     Parameters:
59         students (dict): The dictionary of students.
60         name (str): The name of the student.
61         grade (float): The grade to add (0-100).
62
63     Returns:
64         None
65     """
66     if name in students:
67         if 0 <= grade <= 100:
68             students[name].append(grade)
69     else:

```

API Documentation

students

display_menu()

add_student()

add_grade()

average()

save_to_file()

load_from_file()

delete_student()

delete_grade()

built with 

```

70         print("Grade must be between 0 and 100.")
71     else:
72         print(f"Student {name} does not exist.")

```

Add a grade to a student's list of grades.

Parameters: students (dict): The dictionary of students. name (str): The name of the student.
grade (float): The grade to add (0-100).

Returns: None

def average(students: Dict[str, List[float]], name: str) -> float | None:

▼ View Source

```

75 def average(students: Dict[str, List[float]], name: str) -> Optional[float]:
76     """
77     Calculate the average grade for a student.
78
79     Parameters:
80         students (dict): The dictionary of students.
81         name (str): The name of the student.
82
83     Returns:
84         float or None: The average grade, or None if no grades exist.
85     """
86     if name in students and students[name]:
87         return sum(students[name]) / len(students[name])
88     else:
89         print(f"{name} has no grades.")
90     return None

```

Calculate the average grade for a student.

Parameters: students (dict): The dictionary of students. name (str): The name of the student.

Returns: float or None: The average grade, or None if no grades exist.

API Documentation

students

display_menu()

add_student()

add_grade()

average()

save_to_file()

load_from_file()

delete_student()

delete_grade()

built with 

def save_to_file(students: Dict[str, List[float]], filename: str) -> None:

[▼ View Source](#)

```

93  def save_to_file(students: Dict[str, List[float]], filename: str) -> None:
94      """
95      Save the students dictionary to a file in JSON format.
96
97      Parameters:
98          students (dict): The dictionary of students.
99          filename (str): The name of the file to save to.
100
101      Returns:
102          None
103      """
104      with open(filename, "w") as file:
105          json.dump(students, file)

```

Save the students dictionary to a file in JSON format.

Parameters: students (dict): The dictionary of students. filename (str): The name of the file to save to.

Returns: None

def load_from_file(filename: str) -> Dict[str, List[float]]:

[▼ View Source](#)

```

108  def load_from_file(filename: str) -> Dict[str, List[float]]:
109      """
110      Load the students dictionary from a file in JSON format.
111
112      Parameters:
113          filename (str): The name of the file to load from.
114
115      Returns:
116          dict: The dictionary of students loaded from the file, or an empty dict.
117      """
118      if not os.path.exists(filename):

```

API Documentation

students

display_menu()

add_student()

add_grade()

average()

save_to_file()

load_from_file()

delete_student()

delete_grade()

built with 

undocumented API documentation

```

119         print("File does not exist.")
120         return {}
121
122     with open(filename, "r") as file:
123         return json.load(file)

```

Load the students dictionary from a file in JSON format.

Parameters: filename (str): The name of the file to load from.

Returns: dict: The dictionary of students loaded from the file, or an empty dictionary if the file does not exist.

def delete_student(students: Dict[str, List[float]], name: str) -> None: [▼ View Source](#)

```

126 def delete_student(students: Dict[str, List[float]], name: str) -> None:
127     """
128     Delete a student from the students dictionary.
129
130     Parameters:
131         students (dict): The dictionary of students.
132         name (str): The name of the student to delete.
133
134     Returns:
135         None
136     """
137     if name in students:
138         del students[name]
139         print(f"Deleted student {name}.")
140     else:
141         print(f"Student {name} does not exist.")

```

Delete a student from the students dictionary.

Parameters: students (dict): The dictionary of students. name (str): The name of the student to delete.

API Documentation

students

display_menu()

add_student()

add_grade()

average()

save_to_file()

load_from_file()

delete_student()

delete_grade()

built with 

Returns: None

def delete_grade(students: Dict[str, List[float]], name: str, grade: float) -> **None**:

▼ View Source

```

144 def delete_grade(students: Dict[str, List[float]], name: str, grade: float) -> None:
145     """
146     Delete a grade from a student's list of grades.
147
148     Parameters:
149         students (dict): The dictionary of students.
150         name (str): The name of the student.
151         grade (float): The grade to delete.
152
153     Returns:
154         None
155     """
156     if name not in students:
157         print(f"Student {name} does not exist.")
158         return
159     if grade not in students[name]:
160         print(f"{name} does not have grade {grade}.")
161         return
162     students[name].remove(grade)
163     print(f"Removed grade {grade} from {name}.")

```

Delete a grade from a student's list of grades.

Parameters: students (dict): The dictionary of students. name (str): The name of the student.
grade (float): The grade to delete.

Returns: None