Project 1

Project Overview: Student Record Management System

- **Level:** Easy
- **Language:** Python (Command-Line Interface)

Key Features:

1. **Add New Student:**
   Collects roll number, name, and marks from the user and saves it to the data list.
2. **View All Students:**
   Displays all student records stored in the system.
3. **Search by Roll Number:**
   Finds and displays a student record by their unique roll number.
4. **Update Student Record:**
   Allows updating the name and marks of an existing student using their roll number.
5. **Delete Student Record:**
   Deletes a student record from the system using the roll number.
6. **Save Records to File:**
   Saves the current student data to a JSON file (students.json), ensuring data is not lost when the program exits.
7. **Exit:**
   Safely exits the program.

How the Program Works:

- **The program stores student records as a list of dictionaries.**
- **When the program starts, it loads the existing records from students.json if the file is present.**
- **The user is repeatedly presented with a menu to choose operations until they choose to exit.**
- **Data is saved to the JSON file when the user selects the "Save" option.**

Technical Concepts Used:

- **File Handling: Using JSON for persistent storage.**
- **Lists and Dictionaries: For storing and managing student records.**

- **Functions: For modular, reusable code.**
- **Loops and Conditionals: To display menus and process user input.**
- **Input/Output: For user interaction.**

Advantages:

- **Very beginner-friendly.**
- **Easy to understand Python basics like lists, dictionaries, file operations, and functions.**
- **Can be expanded with additional features later (like sorting, validation, etc.).**

Improvement Suggestions:

1. **Input Validation:**
    - **Ensure that roll numbers are unique.**
    - **Handle invalid input (like entering text instead of numbers).**
2. **Auto Save:**
    - **Save automatically after every add, update, or delete to reduce the chance of unsaved changes.**
3. **UI Improvement:**
    - **Display better formatted tables (maybe using tabulate library later).**
4. **Better File Error Handling:**
    - **Handle cases where the JSON file might be corrupted.**
5. **Data Backup:**
    - **Create a backup file when saving changes.**
6. **Optional: Sorting Feature**
    - **Allow sorting students by name, roll number, or marks.**

Python Code:

```
import os
import json
DATA_FILE = "students.json"
def load_data():
if os.path.exists(DATA_FILE):
with open(DATA_FILE, "r") as f:
return json.load(f)
return []
def save_data(data):
with open(DATA_FILE, "w") as f:
json.dump(data, f, indent=4)
def add_student(data):
roll = input("Enter Roll Number: ")
name = input("Enter Name: ")
marks = float(input("Enter Marks: "))
data.append({"roll": roll, "name": name, "marks": marks})
print("Student added successfully!")
def view_students(data):
if not data:
print("No records found.")
return
for stu in data:
print(f"Roll: {stu['roll']}, Name: {stu['name']}, Marks: {stu['marks']}")
def search_student(data):
roll = input("Enter Roll Number to search: ")
for stu in data:
if stu['roll'] == roll:
print(f"Found: {stu}")
```

```
return
print("Student not found.")
def update_student(data):
roll = input("Enter Roll Number to update: ")
for stu in data:
if stu['roll'] == roll:
stu['name'] = input("Enter new name: ")
stu['marks'] = float(input("Enter new marks: "))
print("Record updated.")
return
print("Student not found.")
def delete_student(data):
roll = input("Enter Roll Number to delete: ")
for i, stu in enumerate(data):
if stu['roll'] == roll:
del data[i]
print("Record deleted.")
return
print("Student not found.")
def main():
data = load_data()
while True:
print("\n--- Student Record System ---")
print("1. Add Student\n2. View All\n3. Search\n4. Update\n5. Delete\n6. Save\n7. Exit")
choice = input("Enter choice: ")
if choice == '1':
add_student(data)
elif choice == '2':
view_students(data)
```

```
elif choice == '3':
search_student(data)
elif choice == '4':
update_student(data)
elif choice == '5':
delete_student(data)
elif choice == '6':
save_data(data)
print("Data saved.")
elif choice == '7':
break
else:
print("Invalid choice.")
if __name__ == "__main__":
main()
```

Output

```
--- Student Record System ---
1. Add Student
2. View All
3. Search
4. Update
5. Delete
6. Save
7. Exit
Enter choice: 1
Enter Roll Number: 101
Enter Name: kavya
Enter Marks: 95.5
Student added successfully!

--- Student Record System ---
1. Add Student
2. View All
3. Search
4. Update
5. Delete
6. Save
7. Exit
Enter choice: 2
Roll: 101, Name: kavya, Marks: 95.5
```

```
--- Student Record System ---
1. Add Student
2. View All
3. Search
4. Update
5. Delete
6. Save
7. Exit
Enter choice: 3
Enter Roll Number to search: 101
Found: {'roll': '101', 'name': 'kavya', 'marks': 95.5}

--- Student Record System ---
1. Add Student
2. View All
3. Search
4. Update
5. Delete
6. Save
7. Exit
Enter choice: 4
Enter Roll Number to update: 101
Enter new name: kavya j
Enter new marks: 98
Record updated.


--- Student Record System ---
1. Add Student
2. View All
3. Search
4. Update
5. Delete
6. Save
7. Exit
Enter choice: 2
Roll: 101, Name: kavya j, Marks: 98.0

--- Student Record System ---
1. Add Student
2. View All
3. Search
4. Update
5. Delete
6. Save
7. Exit
Enter choice: 5
Enter Roll Number to delete: 101
Record deleted.
```

```
--- Student Record System ---
1. Add Student
2. View All
3. Search
4. Update
5. Delete
6. Save
7. Exit
Enter choice: 2
```