NEW PROJECT
WARISHA EMAN
DATA STRUCTURE
BSCS 3(A)
14671

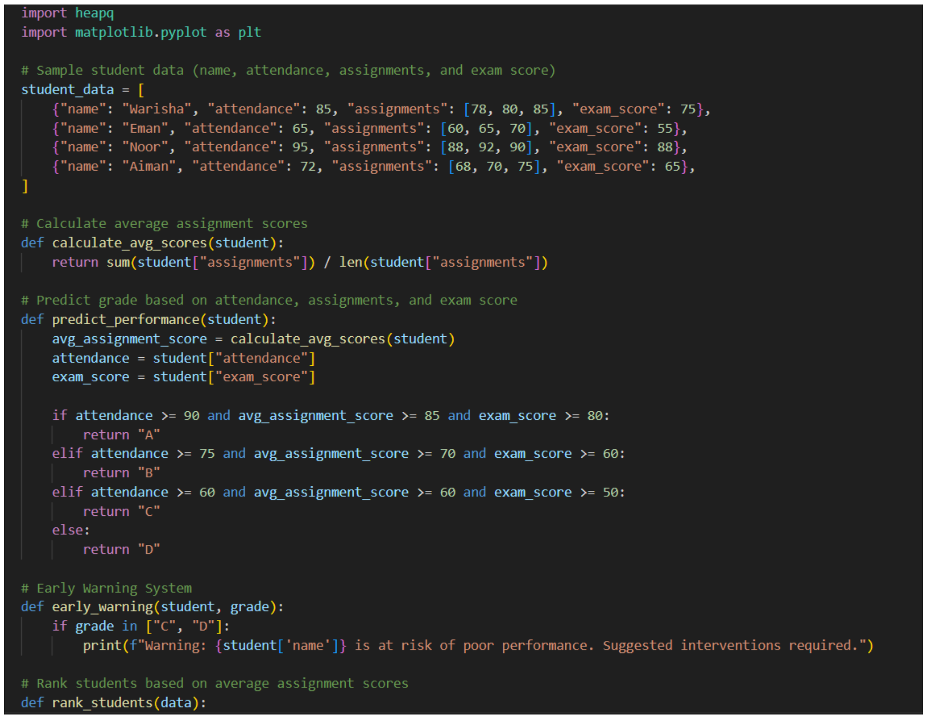
https://github.com/warishaeman28/new-project.git

[Document subtitle]

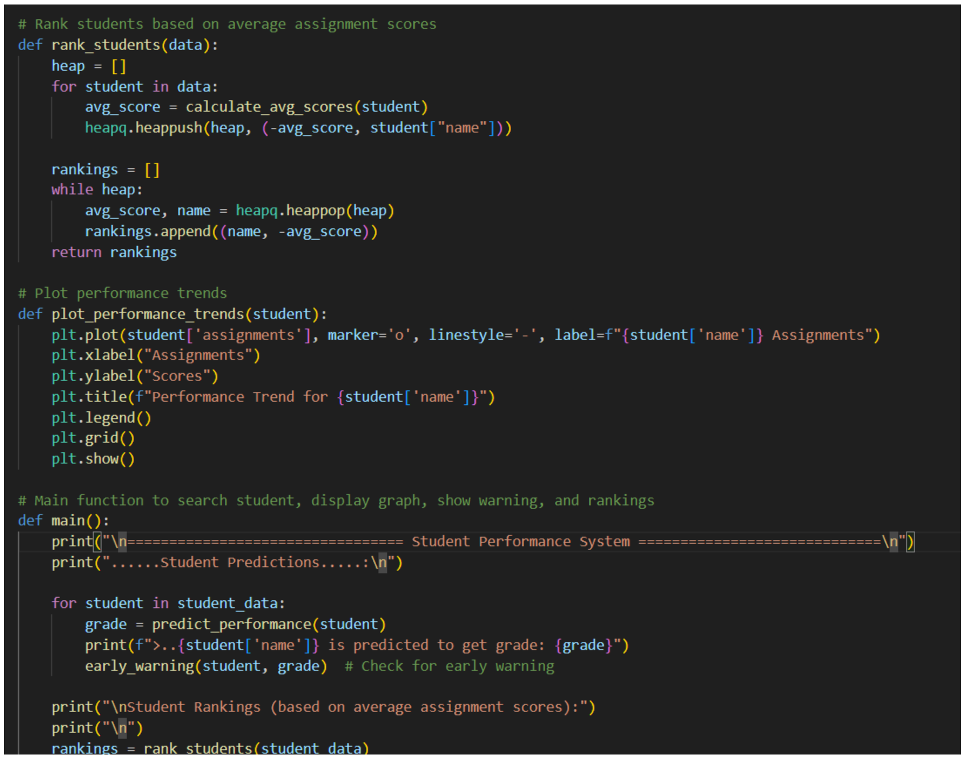
https://github.com/warishaeman28/new-project.git

AI-Based Academic Performance Predictor

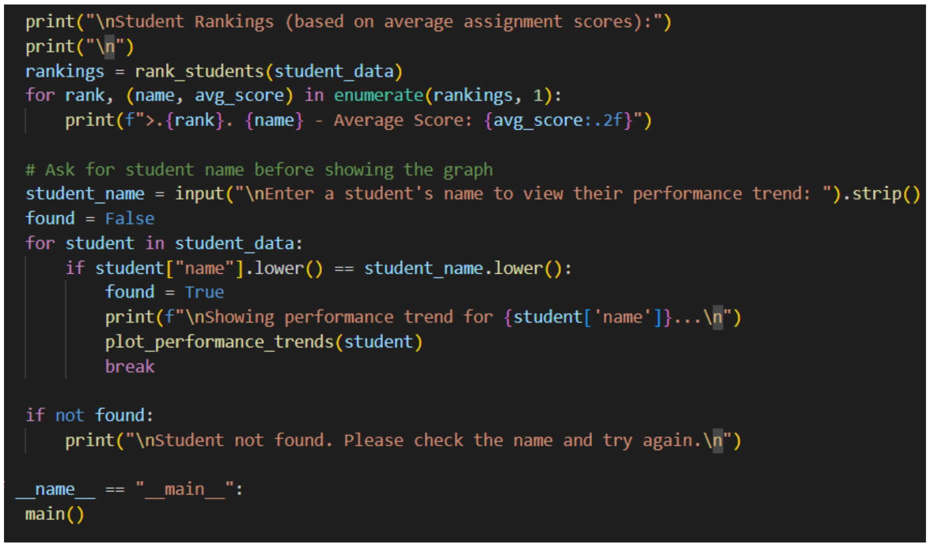
code



```python
import heapq
import matplotlib.pyplot as plt

# Sample student data (name, attendance, assignments, and exam score)
student_data = [
    {"name": "Warisha", "attendance": 85, "assignments": [78, 80, 85], "exam_score": 75},
    {"name": "Eman", "attendance": 65, "assignments": [60, 65, 70], "exam_score": 55},
    {"name": "Noor", "attendance": 95, "assignments": [88, 92, 90], "exam_score": 88},
    {"name": "Aiman", "attendance": 72, "assignments": [68, 70, 75], "exam_score": 65},
]

# Calculate average assignment scores
def calculate_avg_scores(student):
    return sum(student["assignments"]) / len(student["assignments"])

# Predict grade based on attendance, assignments, and exam score
def predict_performance(student):
    avg_assignment_score = calculate_avg_scores(student)
    attendance = student["attendance"]
    exam_score = student["exam_score"]

    if attendance >= 90 and avg_assignment_score >= 85 and exam_score >= 80:
        return "A"
    elif attendance >= 75 and avg_assignment_score >= 70 and exam_score >= 60:
        return "B"
    elif attendance >= 60 and avg_assignment_score >= 60 and exam_score >= 50:
        return "C"
    else:
        return "D"

# Early Warning System
def early_warning(student, grade):
    if grade in ["C", "D"]:
        print(f"Warning: {student['name']} is at risk of poor performance. Suggested interventions required.")

# Rank students based on average assignment scores
def rank_students(data):
```

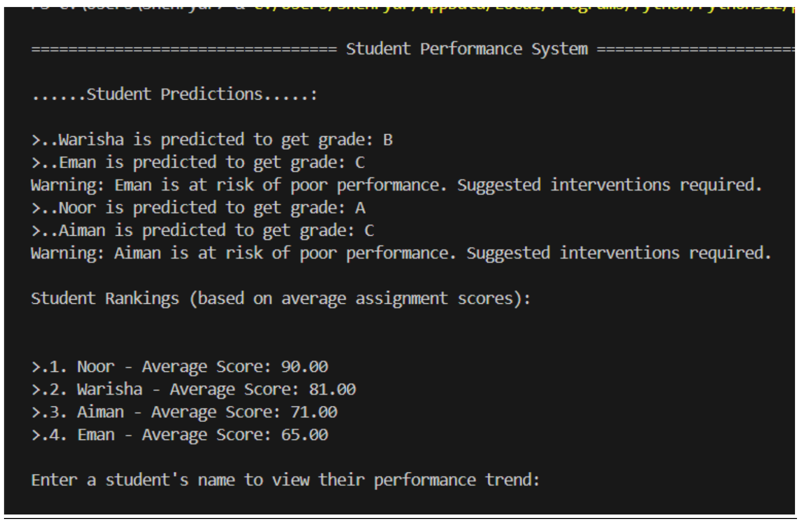


```python
# Rank students based on average assignment scores
def rank_students(data):
    heap = []
    for student in data:
        avg_score = calculate_avg_scores(student)
        heapq.heappush(heap, (-avg_score, student["name"]))

    rankings = []
    while heap:
        avg_score, name = heapq.heappop(heap)
        rankings.append((name, -avg_score))
    return rankings

# Plot performance trends
def plot_performance_trends(student):
    plt.plot(student['assignments'], marker='o', linestyle='-', label=f"{student['name']} Assignments")
    plt.xlabel("Assignments")
    plt.ylabel("Scores")
    plt.title(f"Performance Trend for {student['name']}")
    plt.legend()
    plt.grid()
    plt.show()

# Main function to search student, display graph, show warning, and rankings
def main():
    print("\n================================ Student Performance System ============================\n")
    print("......Student Predictions.....:\n")

    for student in student_data:
        grade = predict_performance(student)
        print(f">..{student['name']} is predicted to get grade: {grade}")
        early_warning(student, grade)  # Check for early warning

    print("\nStudent Rankings (based on average assignment scores):")
    print("\n")
    rankings = rank_students(student_data)
```

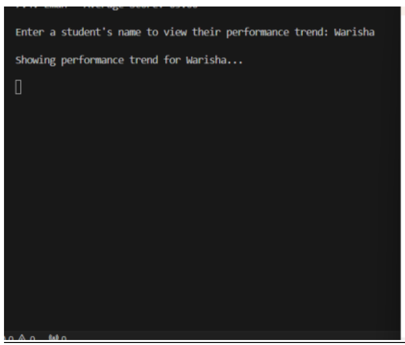


```python
print("\nStudent Rankings (based on average assignment scores):")
print("\n")
rankings = rank_students(student_data)
for rank, (name, avg_score) in enumerate(rankings, 1):
    print(f">.{rank}. {name} - Average Score: {avg_score:.2f}")

# Ask for student name before showing the graph
student_name = input("\nEnter a student's name to view their performance trend: ").strip()
found = False
for student in student_data:
    if student["name"].lower() == student_name.lower():
        found = True
        print(f"\nShowing performance trend for {student['name']}...\n")
        plot_performance_trends(student)
        break

if not found:
    print("\nStudent not found. Please check the name and try again.\n")

__name__ == "__main__":
main()
```

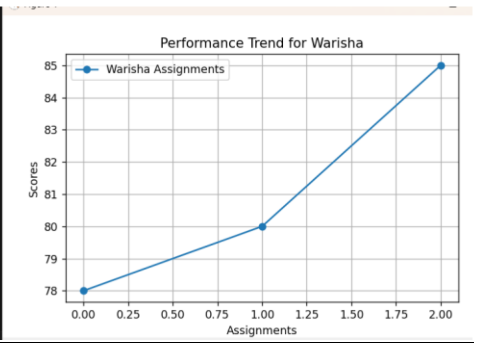
Output



```
================================= Student Performance System =====================

......Student Predictions.....:

>..Warisha is predicted to get grade: B
>..Eman is predicted to get grade: C
Warning: Eman is at risk of poor performance. Suggested interventions required.
>..Noor is predicted to get grade: A
>..Aiman is predicted to get grade: C
Warning: Aiman is at risk of poor performance. Suggested interventions required.

Student Rankings (based on average assignment scores):


>.1. Noor - Average Score: 90.00
>.2. Warisha - Average Score: 81.00
>.3. Aiman - Average Score: 71.00
>.4. Eman - Average Score: 65.00

Enter a student's name to view their performance trend:
```



```
Enter a student's name to view their performance trend: Warisha

Showing performance trend for Warisha...

```


Performance Trend for Warisha
Warisha Assignments
Scores
Assignments