School Student Management — Solution

This PDF contains a ready-to-run solution (no functions) for the School Student Management program, implementing the required 15 steps: variables, data types, numbers, casting, strings, booleans, lists, tuples, sets, dictionaries, if-else, match, and while loop.

Solution Code:

```
# School Student Management System - Solution (No functions)

# 1. Basic variables
school_name = "Green Valley High"
max_students = 30
welcome_msg = f"Welcome to {school_name} Student Management System!"

print(welcome_msg)

# 2. Subjects tuple
subjects = ("Math", "Science", "English")
print("Subjects:", subjects)

# 3. Storage containers
student_ids = set()   # to prevent duplicate IDs
students = []         # list of student dictionaries

# 4-13. Main menu loop with match
while True:
    print("\nMain Menu:")
    print("1 - Add student")
    print("2 - View all students")
    print("3 - Exit and show report")
    choice = input("Enter choice (1/2/3): ").strip()

    match choice:
        case "1":
            # 7. Add student - get ID with validation
            while True:
                sid = input("Enter student ID: ").strip()
                if sid == "":
                    print("ID cannot be empty. Try again.")
                    continue
                if sid in student_ids:
                    print("This ID already exists. Enter a unique ID.")
                else:
                    break

            # 8. Collect name and marks (casting + validation)
            name = input("Enter student full name: ").strip()
            marks = []
            for subj in subjects:
                while True:
                    raw = input(f"Enter marks for {subj} (0-100): ").strip()
                    try:
                        mark = float(raw)
                        if 0 <= mark <= 100:
                            marks.append(mark)
                            break
                        else:
                            print("Marks must be between 0 and 100.")
                    except ValueError:
                        print("Please enter a number.")

            # 9. Calculate average
            average = sum(marks) / len(marks)

            # 10. Determine pass/fail
            is_pass = average >= 50
            status = "Pass" if is_pass else "Fail"

            # 11. Build student dict & store
            student = {
                "id": sid,
                "name": name,
                "grades": marks,
                "average": average,
                "status": status
            }

            if len(students) < max_students:
                students.append(student)
                student_ids.add(sid)
                print(f"Student {name} added. Average: {average:.1f}. Status: {status}.")
            else:
                print("Maximum number of students reached. Cannot add more.")

        case "2":
            # 13. View all students
            if not students:
                print("No students added yet.")
            else:
                print("\nAll Students:")
                for s in students:
                    grades_str = ", ".join(f"{g:.1f}" for g in s["grades"])
                    print(f"ID: {s['id']} | Name: {s['name']} | Grades: [{grades_str}] | Average: {s

        case "3":
            # 14. Exit and final report
            print("\n--- Final Report ---")
            total_students = len(students)
            print(f"Total students added: {total_students}")
            if total_students == 0:
                print("No student data to show.")
            else:
                overall_avg = sum(s["average"] for s in students) / total_students
                for s in students:
                    grades_str = ", ".join(f"{g:.1f}" for g in s["grades"])
                    print(f"ID: {s['id']} | Name: {s['name']} | Grades: [{grades_str}] | Average: {s
                print(f"\nOverall class average: {overall_avg:.1f}")
            break

        case _:
            print("Invalid choice. Please enter 1, 2, or 3.")

# 15. Input validation & UX polish included throughout the code above.
```

Sample Run (example):

```
Sample Run (example interaction):

Welcome to Green Valley High Student Management System!
Subjects: ('Math', 'Science', 'English')

Main Menu:
1 - Add student
2 - View all students
3 - Exit and show report
Enter choice (1/2/3): 1
Enter student ID: S001
Enter student full name: Alice Johnson
Enter marks for Math (0-100): 78
Enter marks for Science (0-100): 82
Enter marks for English (0-100): 69
Student Alice Johnson added. Average: 76.3. Status: Pass.

Main Menu:
1 - Add student
2 - View all students
3 - Exit and show report
Enter choice (1/2/3): 1
Enter student ID: S002
Enter student full name: Bob Smith
Enter marks for Math (0-100): 45
Enter marks for Science (0-100): 52
Enter marks for English (0-100): 48
Student Bob Smith added. Average: 48.3. Status: Fail.
```

```
Main Menu:
1 - Add student
2 - View all students
3 - Exit and show report
Enter choice (1/2/3): 2

All Students:
ID: S001 | Name: Alice Johnson | Grades: [78.0, 82.0, 69.0] | Average: 76.3 | Pass
ID: S002 | Name: Bob Smith | Grades: [45.0, 52.0, 48.0] | Average: 48.3 | Fail

Main Menu:
1 - Add student
2 - View all students
3 - Exit and show report
Enter choice (1/2/3): 3

--- Final Report ---
Total students added: 2
ID: S001 | Name: Alice Johnson | Grades: [78.0, 82.0, 69.0] | Average: 76.3 | Pass
ID: S002 | Name: Bob Smith | Grades: [45.0, 52.0, 48.0] | Average: 48.3 | Fail

Overall class average: 62.3
```