Student Grades Project

Overview:

- *This program implements a Binary Search Tree (BST) to store subjects. Each node in the BST contains a subject name and a sorted linked list of students' grades. The grades are managed using a custom linked list implementation (GradeLinkedList). The program supports various operations on the BST, including insertion, removal, printing, and finding subjects, as well as operations on grades*

Classes:

GradeLinkedList

- **Description**: A linked list implementation to manage grades for a subject.
- **Attributes:**

  GradeNode: Inner class representing a node in the linked list with a grade value and a pointer to the next node.

  ‘Head’: Pointer to the first node in the linked list.
- **Member Functions:**

  GradeLinkedList(): Constructor to initialize an empty linked list.

  InsertGrade(int newGrade): Inserts a new grade into the linked list in a sorted order.

  RemoveGrade(int _grade): Removes a specific grade from the linked list.

  PrintList() const: Prints the linked list.

Classes:

- **BSTNode**

- **Description**: A node in the Binary Search Tree representing a subject with its associated grades.
- **Attributes**:

  subject: Subject name.

  gradeList: Pointer to a GradeLinkedList storing grades.

  left, right: Pointers to the left and right child nodes in the BST.

Continued BSTNode Class:

- **Member Functions:**

  BSTNode(): Default constructor.

  BSTNode(const string& _subject): Constructor with subject initialization.

  insertSubject(string newSubject): Inserts a new subject into the BST.

  insertGradeForSubject(const string& subject, int newGrade): Inserts a new grade for a specific subject.

  printSubjects(): Prints all subjects along with their grades.

  findSubject(const string& _subject): Checks if a subject exists in the BST.

  removeSubject(const string& rsubject): Removes a subject from the BST.

  removeGradeForSubject(const string& subject, int grade): Removes a grade from a specific subject.

  maxGradeInAll(): Finds the maximum grade across all subjects.

Usage:

- **Insert a new subject:**

```
BSTNode* bstNode = new BSTNode();
bstNode->insertSubject(string newSubject);
```

- **Insert a new grade for a subject:**

```
bstNode->insertGradeForSubject(const string& subject, int
newGrade);
```

- **Print all subjects with grades:**

```
bstNode->printSubjects();
```

Usage:

- **Find a subject:**

```
if (bstNode->findSubject(const string& searchSubject) {
    cout << "Subject found." << endl;
} else {
    cout << "Subject not found." << endl;
}
```

- **Remove a subject:**

```
bstNode->removeSubject(const string& subjectToRemove).
```

Usage:

- **Remove a grade from a subject:**

```
bstNode->removeGradeForSubject(const string& subject, int grade);
```

- **Find the maximum grade in all subjects:**

```
int maxGrade = bstNode->maxGradeInAll();
cout << "Maximum grade: " << maxGrade << endl;
```

Extra Notes:

- *Ensure proper memory management, especially for dynamic memory allocated in linked lists.*
- *Handle edge cases such as empty trees or linked lists gracefully.*