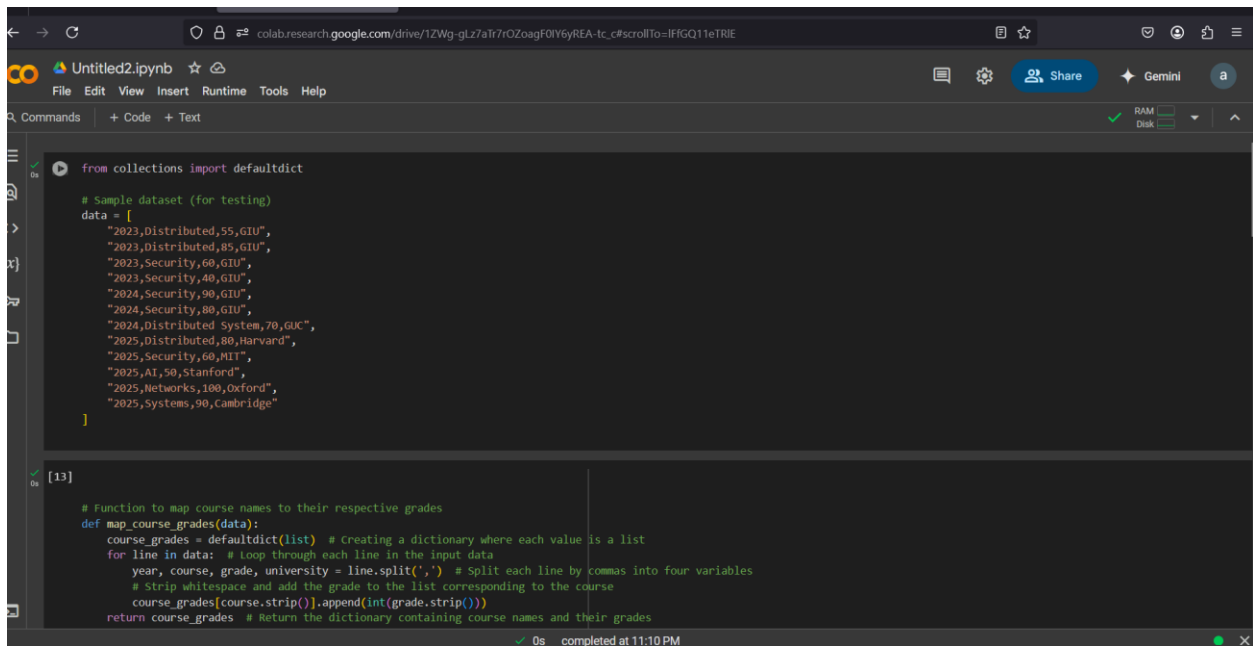


## SCREENSHOTS FOR CODE

### THE DATA:



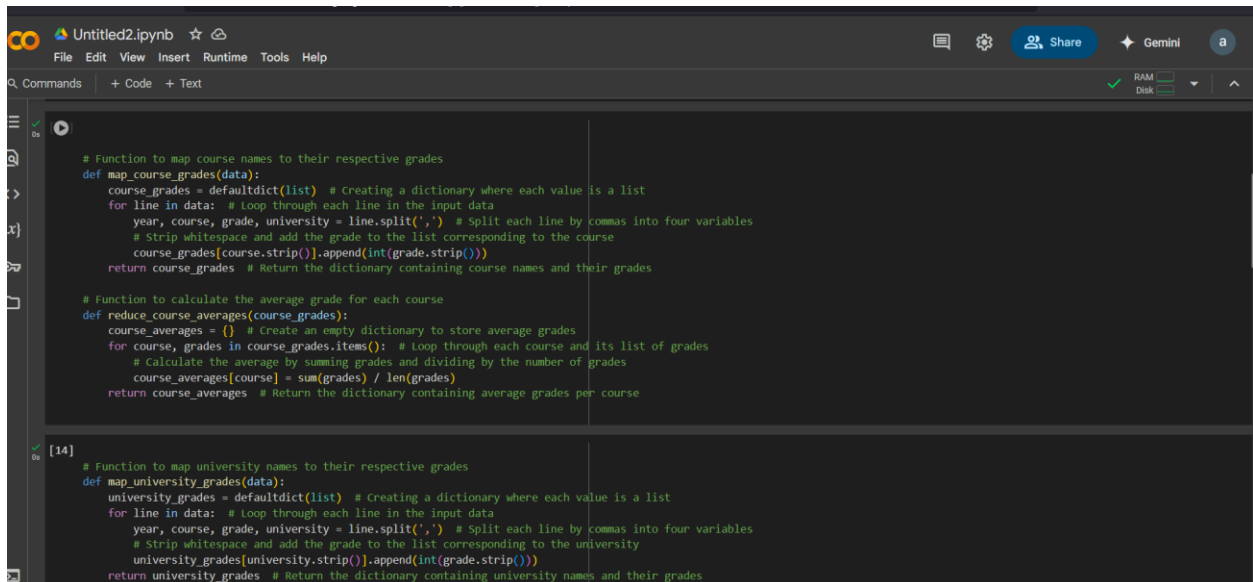
The screenshot shows a Jupyter Notebook interface with a dark theme. The top bar includes the Google Colab logo, the file name 'Untitled2.ipynb', and various icons for file operations, settings, and sharing. The main area displays a Python script. The first cell contains a sample dataset for testing, which is a list of strings representing course information. The second cell contains a function named 'map\_course\_grades' that takes a list of strings and returns a dictionary where each key is a course name and each value is a list of grades. The function uses 'line.split(',')' to parse the input data and 'course.strip()' to clean the course names. The output of the function is shown as a dictionary.

```
from collections import defaultdict

# Sample dataset (for testing)
data = [
    "2023,Distributed,55,GIU",
    "2023,Distributed,85,GIU",
    "2023,Security,60,GIU",
    "2023,Security,40,GIU",
    "2024,Security,90,GIU",
    "2024,Security,80,GIU",
    "2024,Distributed System,70,GUC",
    "2025,Distributed,80,Harvard",
    "2025,Security,60,MIT",
    "2025,AI,50,Stanford",
    "2025,Networks,100,Oxford",
    "2025,Systems,90,Cambridge"
]

# Function to map course names to their respective grades
def map_course_grades(data):
    course_grades = defaultdict(list) # Creating a dictionary where each value is a list
    for line in data: # Loop through each line in the input data
        year, course, grade, university = line.split(',') # Split each line by commas into four variables
        # Strip whitespace and add the grade to the list corresponding to the course
        course_grades[course.strip()].append(int(grade.strip()))
    return course_grades # Return the dictionary containing course names and their grades
```

### TASK 1



The screenshot shows a Jupyter Notebook interface with a dark theme. The top bar includes the Google Colab logo, the file name 'Untitled2.ipynb', and various icons for file operations, settings, and sharing. The main area displays a Python script. The first cell contains a function named 'map\_course\_grades' that takes a list of strings and returns a dictionary where each key is a course name and each value is a list of grades. The second cell contains a function named 'reduce\_course\_averages' that takes a dictionary of course grades and returns a dictionary where each key is a course name and each value is the average grade. The third cell contains a function named 'map\_university\_grades' that takes a list of strings and returns a dictionary where each key is a university name and each value is a list of grades. The function uses 'line.split(',')' to parse the input data and 'university.strip()' to clean the university names.

```
# Function to map course names to their respective grades
def map_course_grades(data):
    course_grades = defaultdict(list) # Creating a dictionary where each value is a list
    for line in data: # Loop through each line in the input data
        year, course, grade, university = line.split(',') # Split each line by commas into four variables
        # Strip whitespace and add the grade to the list corresponding to the course
        course_grades[course.strip()].append(int(grade.strip()))
    return course_grades # Return the dictionary containing course names and their grades

# Function to calculate the average grade for each course
def reduce_course_averages(course_grades):
    course_averages = {} # Create an empty dictionary to store average grades
    for course, grades in course_grades.items(): # Loop through each course and its list of grades
        # Calculate the average by summing grades and dividing by the number of grades
        course_averages[course] = sum(grades) / len(grades)
    return course_averages # Return the dictionary containing average grades per course

# Function to map university names to their respective grades
def map_university_grades(data):
    university_grades = defaultdict(list) # Creating a dictionary where each value is a list
    for line in data: # Loop through each line in the input data
        year, course, grade, university = line.split(',') # Split each line by commas into four variables
        # Strip whitespace and add the grade to the list corresponding to the university
        university_grades[university.strip()].append(int(grade.strip()))
    return university_grades # Return the dictionary containing university names and their grades
```

### TASK 2

```
Untitled2.ipynb
File Edit View Insert Runtime Tools Help
Q Commands + Code + Text
RAM Disk

[14]
# Function to map university names to their respective grades
def map_university_grades(data):
    university_grades = defaultdict(list) # Creating a dictionary where each value is a list
    for line in data: # Loop through each line in the input data
        year, course, grade, university = line.split(',') # Split each line by commas into four variables
        # Strip whitespace and add the grade to the list corresponding to the university
        university_grades[university.strip()].append(int(grade.strip()))
    return university_grades # Return the dictionary containing university names and their grades

# Function to calculate the average grade for each university
def reduce_university_averages(university_grades):
    university_averages = {} # Create an empty dictionary to store average grades
    for university, grades in university_grades.items(): # Loop through each university and its list of grades
        # Calculate the average by summing grades and dividing by the number of grades
        university_averages[university] = sum(grades) / len(grades)
    return university_averages # Return the dictionary containing average grades per university

[15]
# Function to map academic years to their respective grades
def map_year_grades(data):
    year_grades = defaultdict(list) # Creating a dictionary where each value is a list
    for line in data: # Loop through each line in the input data
        year, course, grade, university = line.split(',') # Split each line by commas into four variables
        # Strip whitespace and add the grade to the list corresponding to the year
        year_grades[year.strip()].append(int(grade.strip()))
    return year_grades # Return the dictionary containing year names and their grades
```

## Bounes

```
Untitled2.ipynb
File Edit View Insert Runtime Tools Help
Q Commands + Code + Text
RAM Disk

[15]
# Function to map academic years to their respective grades
def map_year_grades(data):
    year_grades = defaultdict(list) # Creating a dictionary where each value is a list
    for line in data: # Loop through each line in the input data
        year, course, grade, university = line.split(',') # Split each line by commas into four variables
        # Strip whitespace and add the grade to the list corresponding to the year
        year_grades[year.strip()].append(int(grade.strip()))
    return year_grades # Return the dictionary containing year names and their grades

# Function to get the top 3 highest grades per academic year
def reduce_top_3_grades(year_grades):
    top_grades = {} # Create an empty dictionary to store the top 3 grades per year
    for year, grades in year_grades.items(): # Loop through each year and its list of grades
        # Sort the grades in descending order and take the top 3 grades
        top_grades[year] = sorted(grades, reverse=True)[:3]
    return top_grades # Return the dictionary containing the top 3 grades per year

if __name__ == "__main__": # Entry point for the script
    # Task 1 Output - Calculate average grade per course
    course_grades = map_course_grades(data) # Map the course names to their grades using the map function
    course_averages = reduce_course_averages(course_grades) # Reduce the mapped data to calculate the average grade per course
    print("Average Grade per Course:") # Print a heading for course average grades
    for course, avg in course_averages.items(): # Iterate through the calculated average grades
        print(f"{course}: {avg:.2f}") # Print each course name and its average grade formatted to 2 decimal places

    # Task 2 Output - Calculate average grade per university
    year_grades = map_year_grades(data) # Map the year names to their grades using the map function
    year_averages = reduce_year_averages(year_grades) # Reduce the mapped data to calculate the average grade per year
    print("Average Grade per Year:") # Print a heading for year average grades
    for year, avg in year_averages.items(): # Iterate through the calculated average grades
        print(f"{year}: {avg:.2f}") # Print each year name and its average grade formatted to 2 decimal places

    # Task 3 Output - Get the top 3 highest grades per academic year
    top_grades = reduce_top_3_grades(year_grades) # Get the top 3 highest grades per academic year
    for year, grades in top_grades.items(): # Iterate through the top 3 grades per year
        print(f"Top 3 Grades for {year}:") # Print a heading for top 3 grades
        for grade in grades: # Iterate through the top 3 grades
            print(f"{grade}") # Print each grade

0s completed at 11:10 PM
```

## MAIN:

```
Untitled2.ipynb
File Edit View Insert Runtime Tools Help
Q Commands + Code + Text
RAM Disk
if __name__ == "__main__": # Entry point for the script
    # Task 1 Output - Calculate average grade per course
    course_grades = map_course_grades(data) # Map the course names to their grades using the map function
    course_averages = reduce_course_averages(course_grades) # Reduce the mapped data to calculate the average grade per course
    print("Average Grade per Course:") # Print a heading for course average grades
    for course, avg in course_averages.items(): # Iterate through the calculated average grades
        print(f'{course}: {avg:.2f}') # Print each course name and its average grade formatted to 2 decimal places

    # Task 2 Output - Calculate average grade per university
    university_grades = map_university_grades(data) # Map the university names to their grades using the map function
    university_averages = reduce_university_averages(university_grades) # Reduce the mapped data to calculate the average grade per university
    print("\nAverage Grade per University:") # Print a heading for university average grades
    for university, avg in university_averages.items(): # Iterate through the calculated average grades per university
        print(f'{university}: {avg:.2f}') # Print each university name and its average grade formatted to 2 decimal places

    # Bonus Task Output - Get the top 3 grades per year
    year_grades = map_year_grades(data) # Map the academic years to their grades using the map function
    top_3_grades = reduce_top_3_grades(year_grades) # Reduce the mapped data to find the top 3 grades per year
    print("\nTop 3 Grades per Year:") # Print a heading for the top 3 grades per year
    for year, grades in top_3_grades.items(): # Iterate through the calculated top 3 grades per year
        print(f'{year}: {grades}') # Print each year and its list of top 3 grades

Average Grade per Course:
Distributed: 73.33
Security: 66.00
Distributed System: 70.00
AI: 50.00
Networks: 100.00
Systems: 90.00

Average Grade per University:
GIU: 68.33
GLC: 70.00
Harvard: 80.00
MIT: 60.00
Stanford: 50.00
Oxford: 100.00
Cambridge: 90.00

0s completed at 11:10 PM
```

SCREENSHOTS FOR OUTPUT :

The ouput for task 1:

```
colab.research.google.com/drive/1ZWg-gLz7a1r7rOZoaGf0Y6yREA-1c_c#scrollTo=IfIGQ11e1RIE
Untitled2.ipynb
File Edit View Insert Runtime Tools Help
Q Commands + Code + Text
RAM Disk
Average Grade per Course:
Distributed: 73.33
Security: 66.00
Distributed System: 70.00
AI: 50.00
Networks: 100.00
Systems: 90.00
```

OUTPUT FOR TASK 2:

```
Average Grade per University:
GIU: 68.33
GLC: 70.00
Harvard: 80.00
MIT: 60.00
Stanford: 50.00
Oxford: 100.00
Cambridge: 90.00
```

OUTPUT FOR BOUNES :

```
Top 3 Grades per Year:
2023: [85, 60, 55]
2024: [90, 80, 70]
2025: [100, 90, 80]
```

0s completed at 11:10 PM

OUTPUT :

Untitled2.ipynb ☆ ☁

File Edit View Insert Runtime Tools Help

Commands + Code + Text

```
for year in top_3_grades.keys(): # Iterate through the dictionary top_3_grades per year
    print(f"{year}: {grades}") # Print each year and its list of top 3 grades
```

Average Grade per Course:  
Distributed: 73.33  
Security: 66.00  
Distributed System: 70.00  
AI: 50.00  
Networks: 100.00  
Systems: 90.00

Average Grade per University:  
GIU: 68.33  
GUC: 70.00  
Harvard: 80.00  
MIT: 60.00  
Stanford: 50.00  
Oxford: 100.00  
Cambridge: 90.00

Top 3 Grades per Year:  
2023: [85, 60, 55]  
2024: [90, 80, 70]  
2025: [100, 90, 80]

0s completed at 11:10 PM