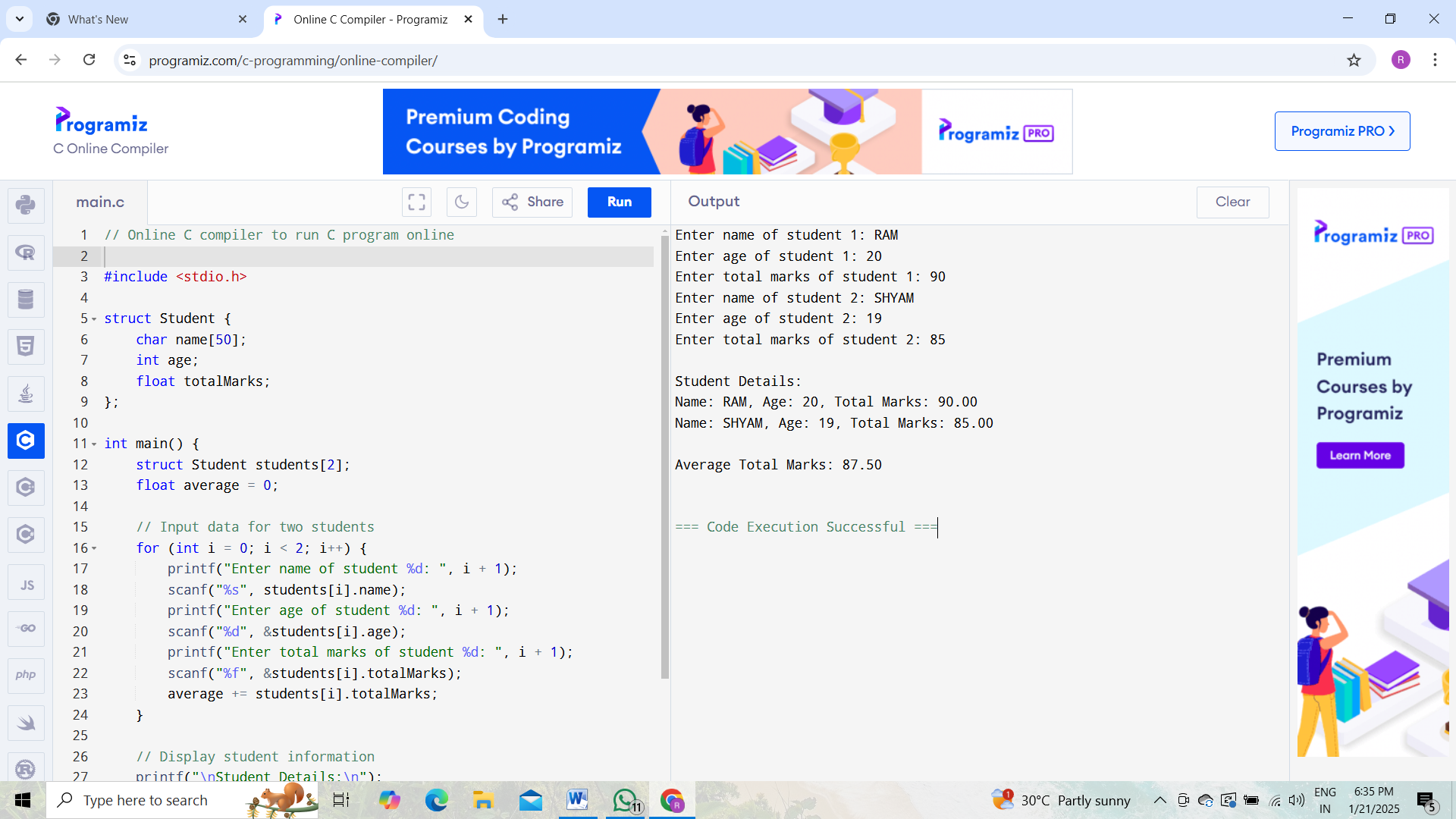
1. **Create a structure called “Student” with member’s name, age , and total marks. Write a C program to input data for two students, display their information, and find the average of total marks**
2. #include <stdio.h>
3. struct Student {
4. char name[50];
5. int age;
6. float totalMarks;
7. };
8. int main() {
9. struct Student students[2];
10. float average = 0;
11. // Input data for two students
12. for (int i = 0; i < 2; i++) {
13. printf("Enter name of student %d: ", i + 1);
14. scanf("%s", students[i].name);
15. printf("Enter age of student %d: ", i + 1);
16. scanf("%d", &students[i].age);
17. printf("Enter total marks of student %d: ", i + 1);
18. scanf("%f", &students[i].totalMarks);
19. average += students[i].totalMarks;
20. }
21. // Display student information
22. printf("\nStudent Details:\n");
23. for (int i = 0; i < 2; i++) {
24. printf("Name: %s, Age: %d, Total Marks: %.2f\n", students[i].name, students[i].age, students[i].totalMarks);
25. }
26. // Calculate and display average marks
27. average /= 2;
28. printf("\nAverage Total Marks: %.2f\n", average);
29. return 0;
30. }

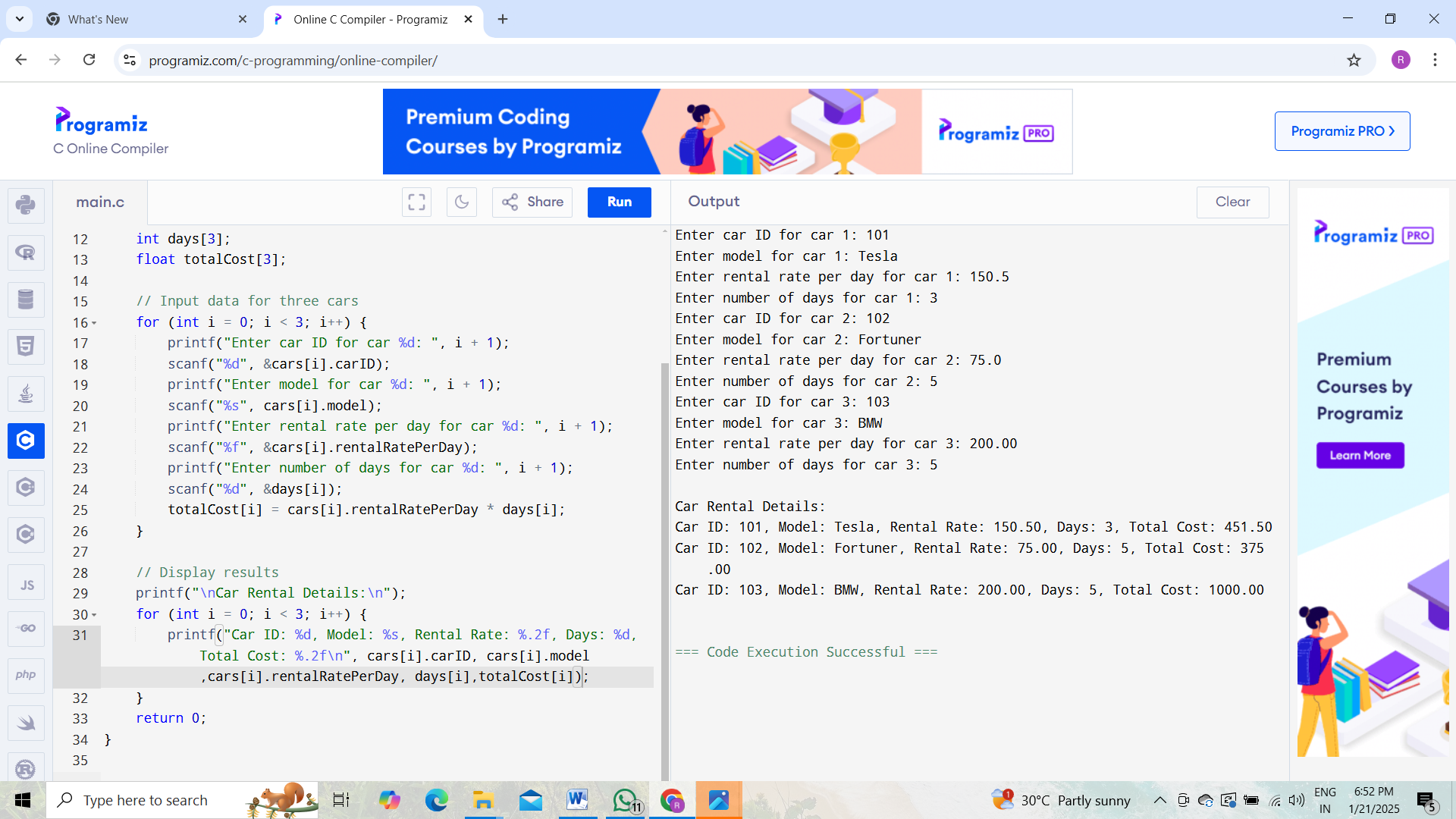
**OUTPUT:**



**2.Design a structure named “Car” to store details like carID, model,, and rental rate per day. Write a C program to input data for three cars, calculate the tSSotal rental cost for a specified number of days, and display the results.**

1. #include <stdio.h>
2. struct Car {
3. int carID;
4. char model[50];
5. float rentalRatePerDay;
6. };
7. int main() {
8. struct Car cars[3];
9. int days[3];
10. float totalCost[3];
11. // Input data for three cars
12. for (int i = 0; i < 3; i++) {
13. printf("Enter car ID for car %d: ", i + 1);
14. scanf("%d", &cars[i].carID);
15. printf("Enter model for car %d: ", i + 1);
16. scanf("%s", cars[i].model);
17. printf("Enter rental rate per day for car %d: ", i + 1);
18. scanf("%f", &cars[i].rentalRatePerDay);
19. printf("Enter number of days for car %d: ", i + 1);
20. scanf("%d", &days[i]);
21. totalCost[i] = cars[i].rentalRatePerDay \* days[i];
22. }
23. // Display results
24. printf("\nCar Rental Details:\n");
25. for (int i = 0; i < 3; i++) {
26. printf("Car ID: %d, Model: %s, Rental Rate: %.2f, Days: %d, Total Cost: %.2f\n",
27. cars[i].carID, cars[i].model, cars[i].rentalRatePerDay, days[i], totalCost[i]);
28. }
29. return 0;
30. }

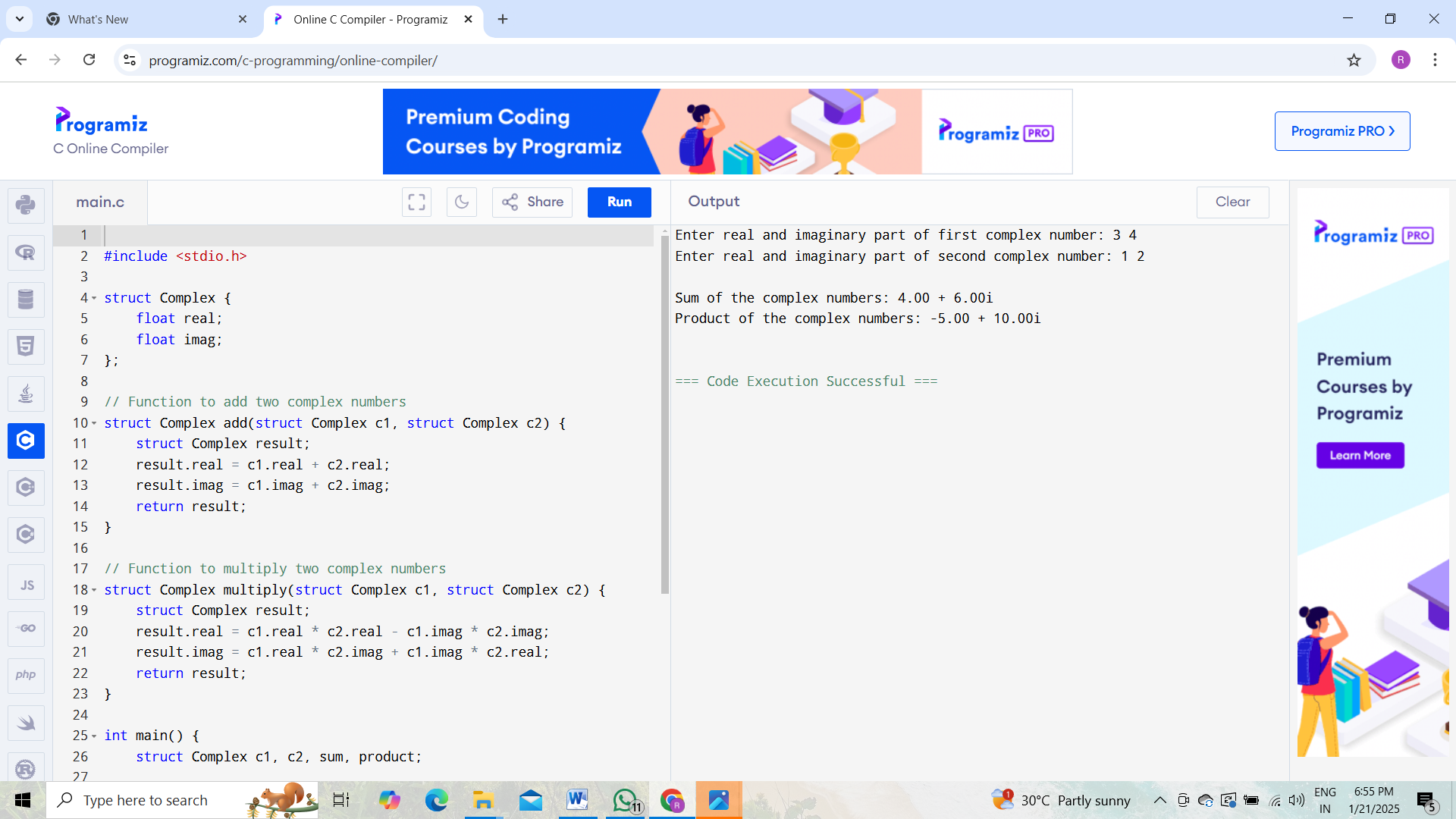
**OUTPUT:**

****

**3) Create a structure named Complex to represent a complex number with real and imaginary parts. Write a C program to add multiply two complex numbers.**

1. #include <stdio.h>
2. struct Complex {
3. float real;
4. float imag;
5. };
6. // Function to add two complex numbers
7. struct Complex add(struct Complex c1, struct Complex c2) {
8. struct Complex result;
9. result.real = c1.real + c2.real;
10. result.imag = c1.imag + c2.imag;
11. return result;
12. }
13. // Function to multiply two complex numbers
14. struct Complex multiply(struct Complex c1, struct Complex c2) {
15. struct Complex result;
16. result.real = c1.real \* c2.real - c1.imag \* c2.imag;
17. result.imag = c1.real \* c2.imag + c1.imag \* c2.real;
18. return result;
19. }
20. int main() {
21. struct Complex c1, c2, sum, product;
22. // Input two complex numbers
23. printf("Enter real and imaginary part of first complex number: ");
24. scanf("%f %f", &c1.real, &c1.imag);
25. printf("Enter real and imaginary part of second complex number: ");
26. scanf("%f %f", &c2.real, &c2.imag);
27. // Add and multiply
28. sum = add(c1, c2);
29. product = multiply(c1, c2);
30. // Display results
31. printf("\nSum of the complex numbers: %.2f + %.2fi\n", sum.real, sum.imag);
32. printf("Product of the complex numbers: %.2f + %.2fi\n", product.real, product.imag);
33. return 0;
34. }

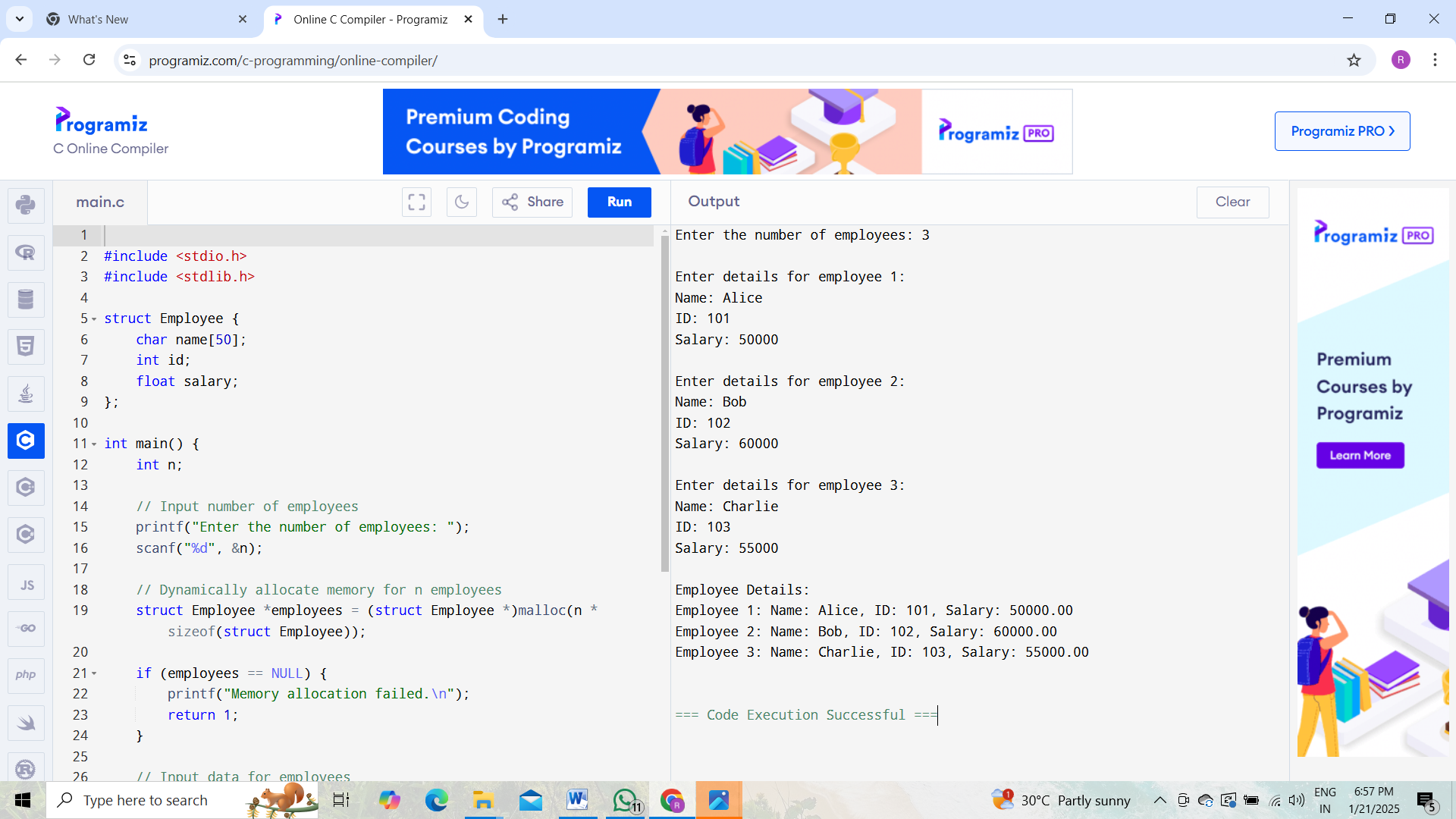
**OUTPUT :**

****

**4) WAP in C to store and print the information of N employees using dynamic memory allocation and structures.**

1. #include <stdio.h>
2. #include <stdlib.h>
3. struct Employee {
4. char name[50];
5. int id;
6. float salary;
7. };
8. int main() {
9. int n;
10. // Input number of employees
11. printf("Enter the number of employees: ");
12. scanf("%d", &n);
13. // Dynamically allocate memory for n employees
14. struct Employee \*employees = (struct Employee \*)malloc(n \* sizeof(struct Employee));
15. if (employees == NULL) {
16. printf("Memory allocation failed.\n");
17. return 1;
18. }
19. // Input data for employees
20. for (int i = 0; i < n; i++) {
21. printf("\nEnter details for employee %d:\n", i + 1);
22. printf("Name: ");
23. scanf("%s", employees[i].name);
24. printf("ID: ");
25. scanf("%d", &employees[i].id);
26. printf("Salary: ");
27. scanf("%f", &employees[i].salary);
28. }
29. // Display employee data
30. printf("\nEmployee Details:\n");
31. for (int i = 0; i < n; i++) {
32. printf("Employee %d: Name: %s, ID: %d, Salary: %.2f\n",
33. i + 1, employees[i].name, employees[i].id, employees[i].salary);
34. }
35. // Free allocated memory
36. free(employees);
37. return 0;
38. }

**OUTPUT:**

****