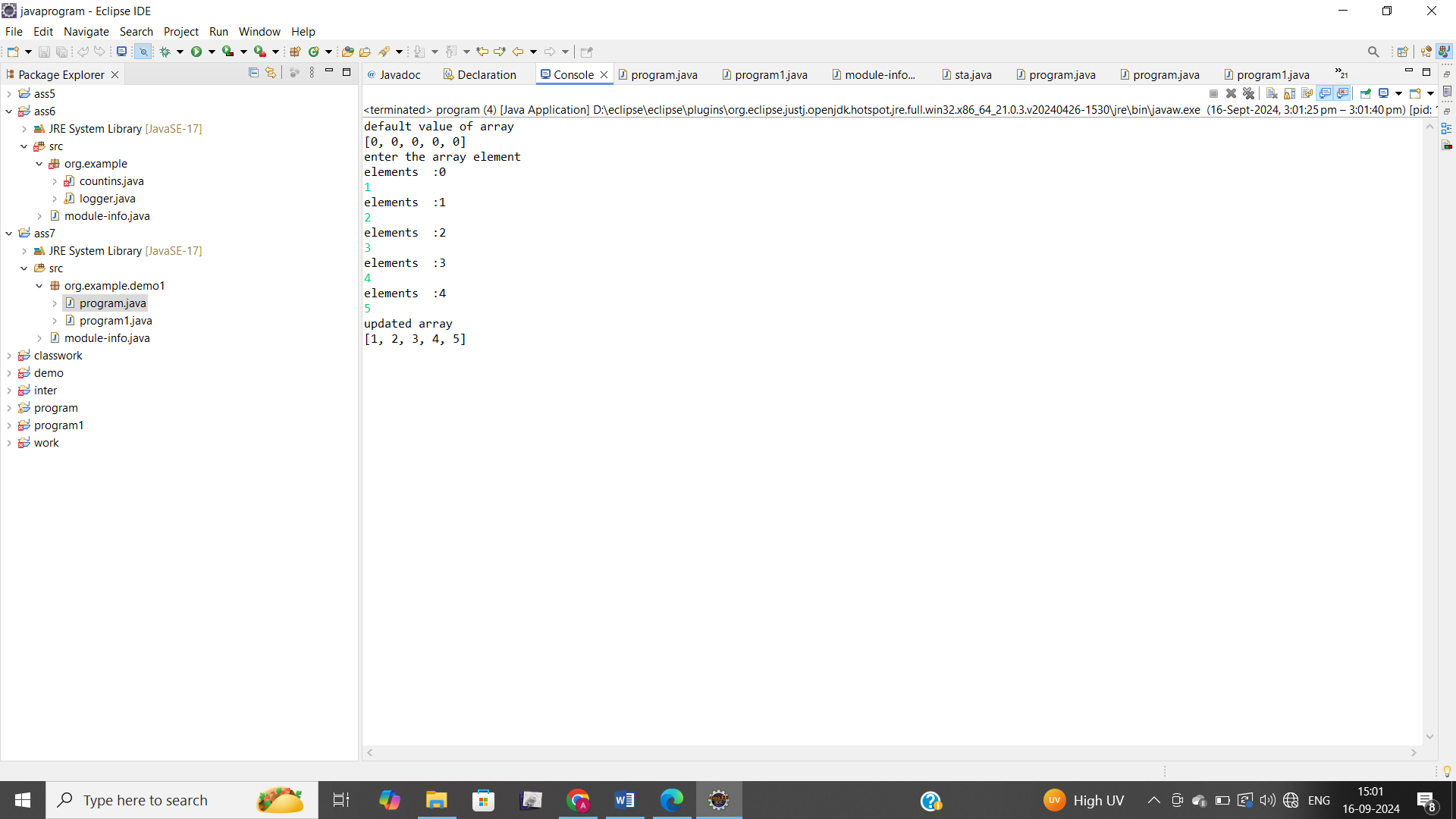
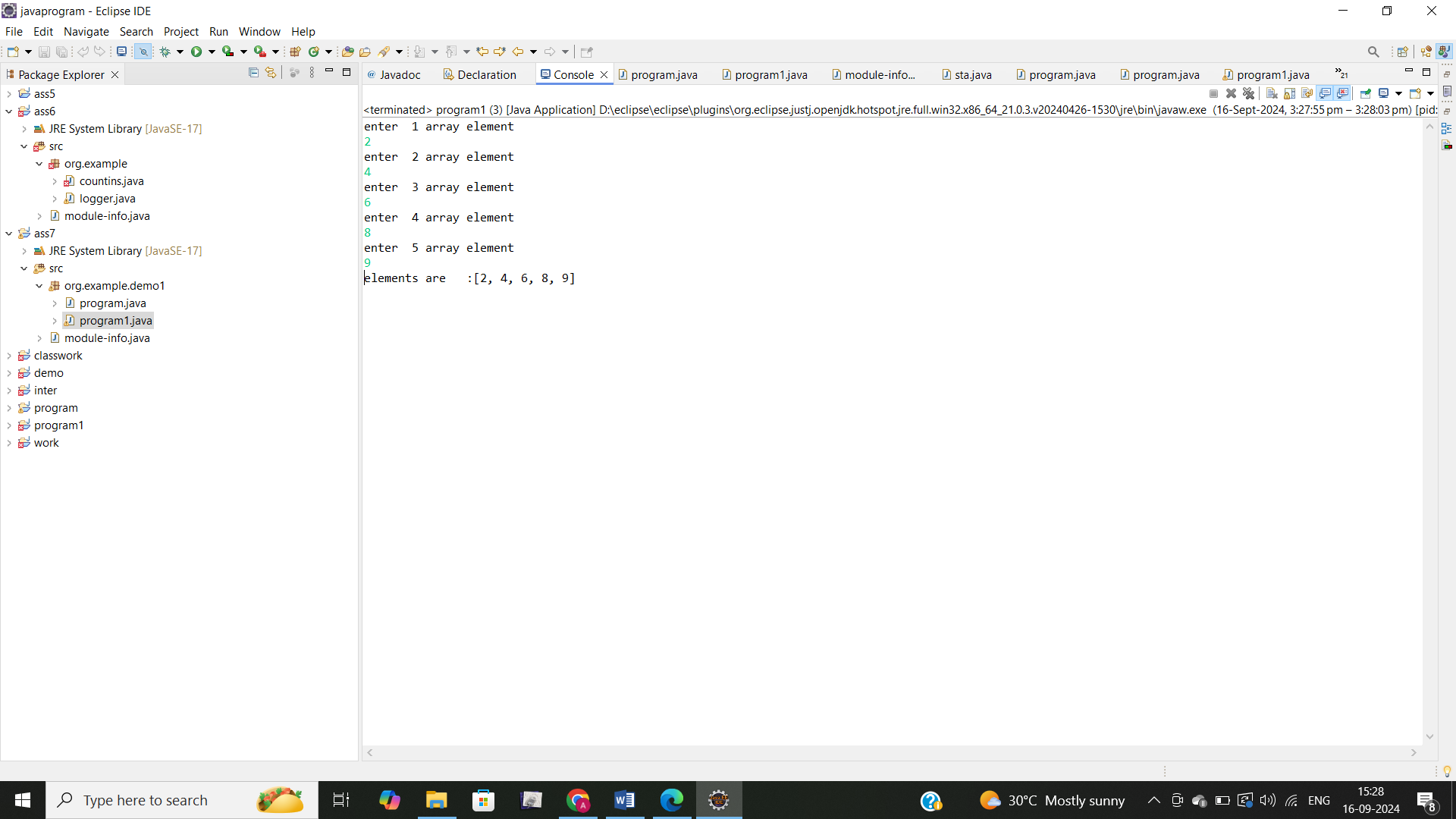
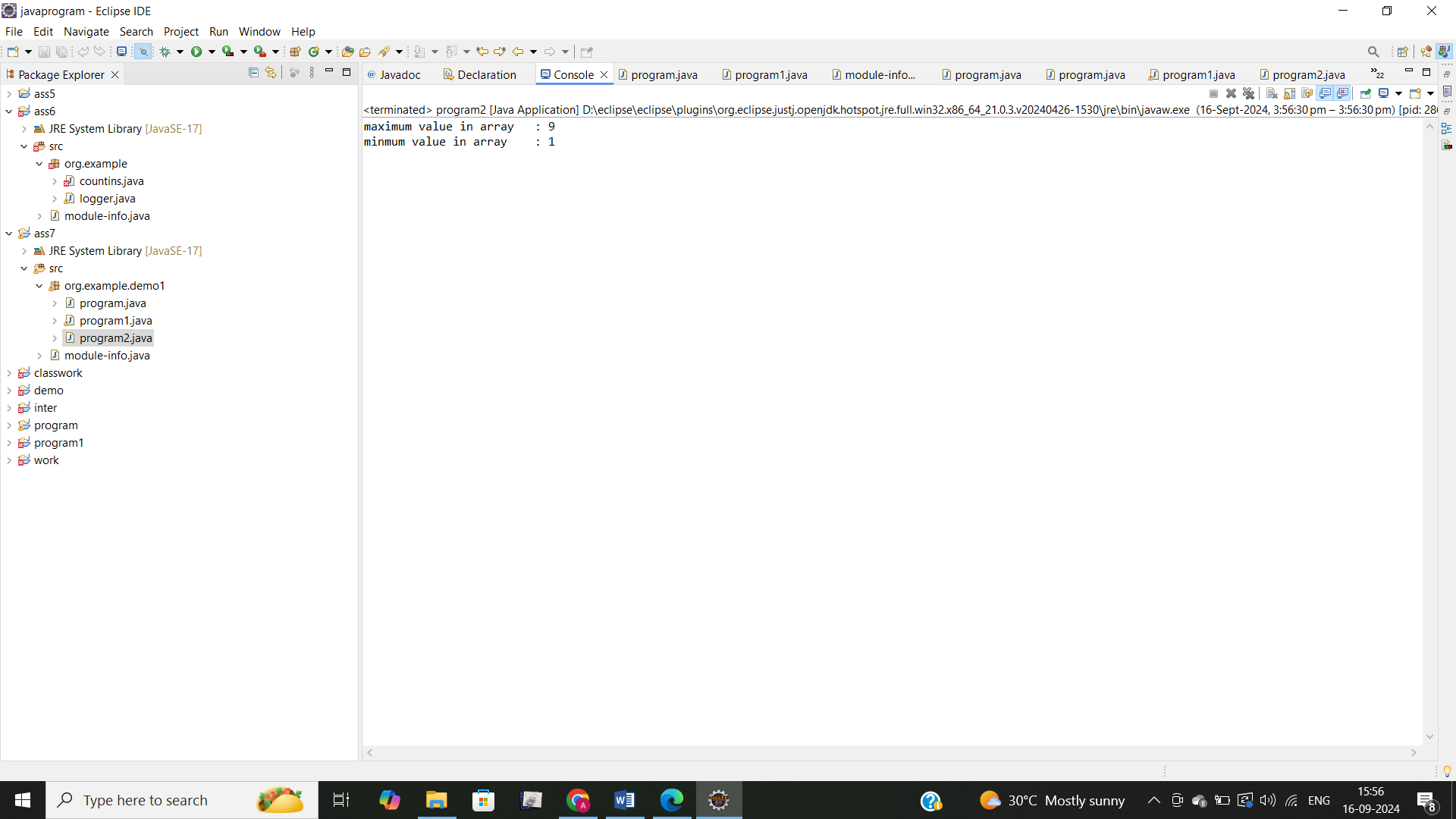
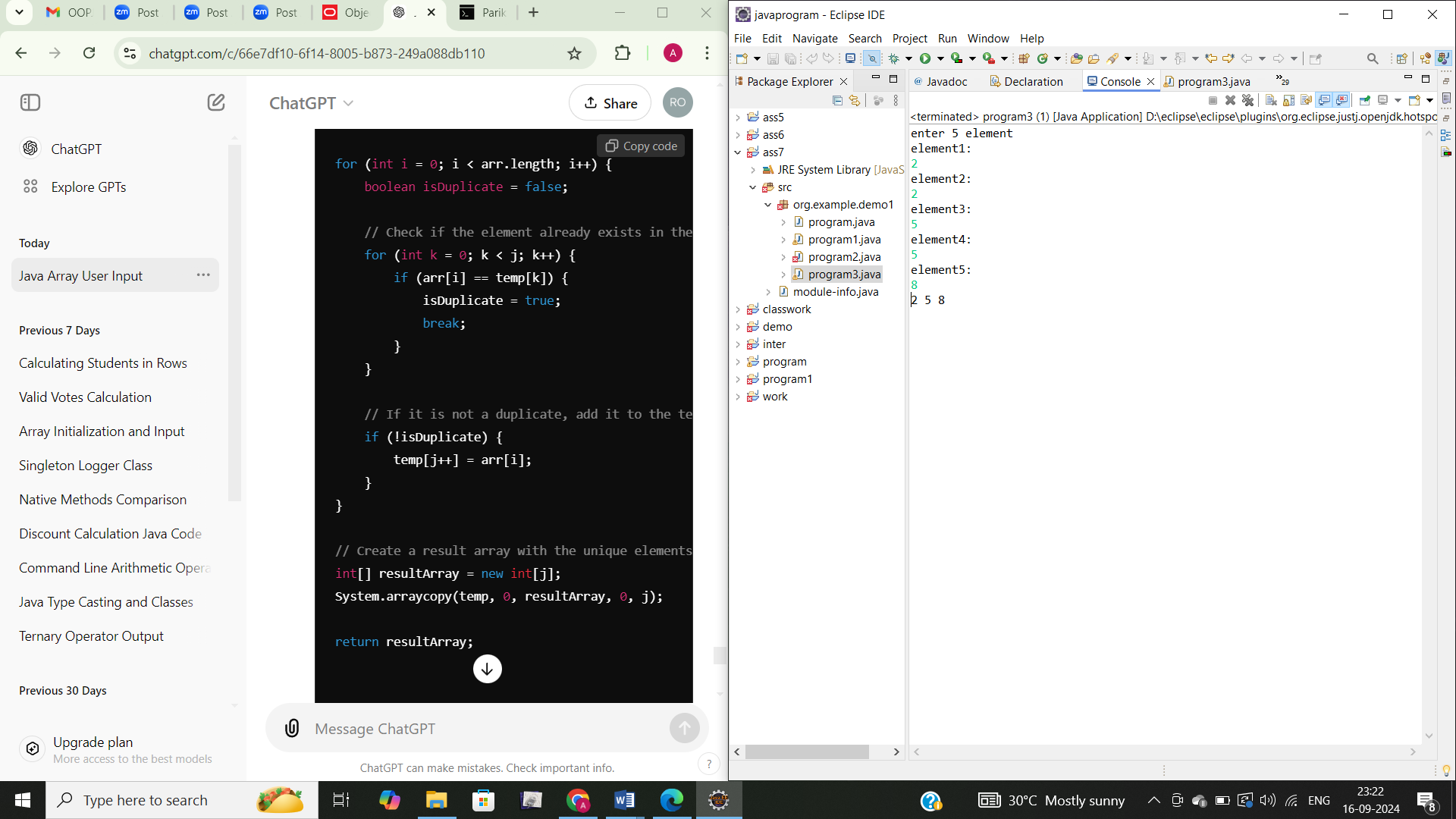
1. Declare a single-dimensional array of 5 integers inside the main method. Traverse the array to print the default values. Then accept records from the user and print the updated values of the array.
2. **package** org.example.demo1;
3. **import** java.util.Scanner;
4. **import** java.util.Arrays;
5. **public** **class** program {
7. **public** **static** **void** main(String[] args) {
9. **int**[] arr=**new** **int**[5];
10. System.***out***.println("default value of array");
11. //for(int index=0;index<arr.length;index++)
12. //System.out.println("elements"+arr[]);
13. System.***out***.println( Arrays.*toString*(arr));
14. Scanner sc=**new** Scanner(System.***in***);
16. System.***out***.println("enter the array element");
17. **for**(**int** index=0;index<arr.length;++index) {
18. System.***out***.println("elements :"+(index));
19. arr[index]=sc.nextInt();
20. }
21. System.***out***.println("updated array ");
22. System.***out***.println(Arrays.*toString*(arr));
23. //for(int index=0;index<arr.length;++index) {
24. // System.out.println("elements :"+arr[index]);
26. sc.close();
28. }

31. }
32. 
33. Declare a single-dimensional array of 5 integers inside the main method. Define a method named acceptRecord to get input from the terminal into the array and another method named printRecord to print the state of the array to the terminal.
34. **package** org.example.demo1;
35. **import** java.util.Scanner;
36. **public** **class** program1 {
37. **public** **static** **void** acceptRecord(**int**[]arr) {
38. Scanner sc=**new** Scanner(System.***in***);
39. **for**(**int** i=0;i<arr.length;i++) {
40. System.***out***.println("enter "+(i+1)+" array element");
41. arr[i]=sc.nextInt();
42. }
43. }
44. **public** **static** **void** printRecord(**int**[]arr) {
45. **for**(**int** i=0;i<arr.length;i++) {
46. System.***out***.println("elemrnts are : "+arr[i]);
47. }
48. }
50. **public** **static** **void** main(String[] args) {
51. **int**[]arr=**new** **int**[5];
52. *acceptRecord*(arr);
53. *printRecord*(arr);
54. }
55. }
56. 
57. Write a program to find the maximum and minimum values in a single-dimensional array of integers.
58. **package** org.example.demo1;
59. **public** **class** program2 {
60. **public** **static** **void** main(String[] args) {
61. **int**[]arr=**new** **int**[]{4,6,1,9,7};
62. **int** max=arr[0];
63. **int** min=arr[0];
64. **for**(**int** i=0;i<arr.length;i++) {
65. **if**(arr[i]>max) {
66. max=arr[i];
68. }
69. **if**(arr[i]<min) {
70. min=arr[i];
71. }
72. }
73. System.***out***.println("maximum value in array : "+max);
74. System.***out***.println("minmum value in array : "+min);
76. }

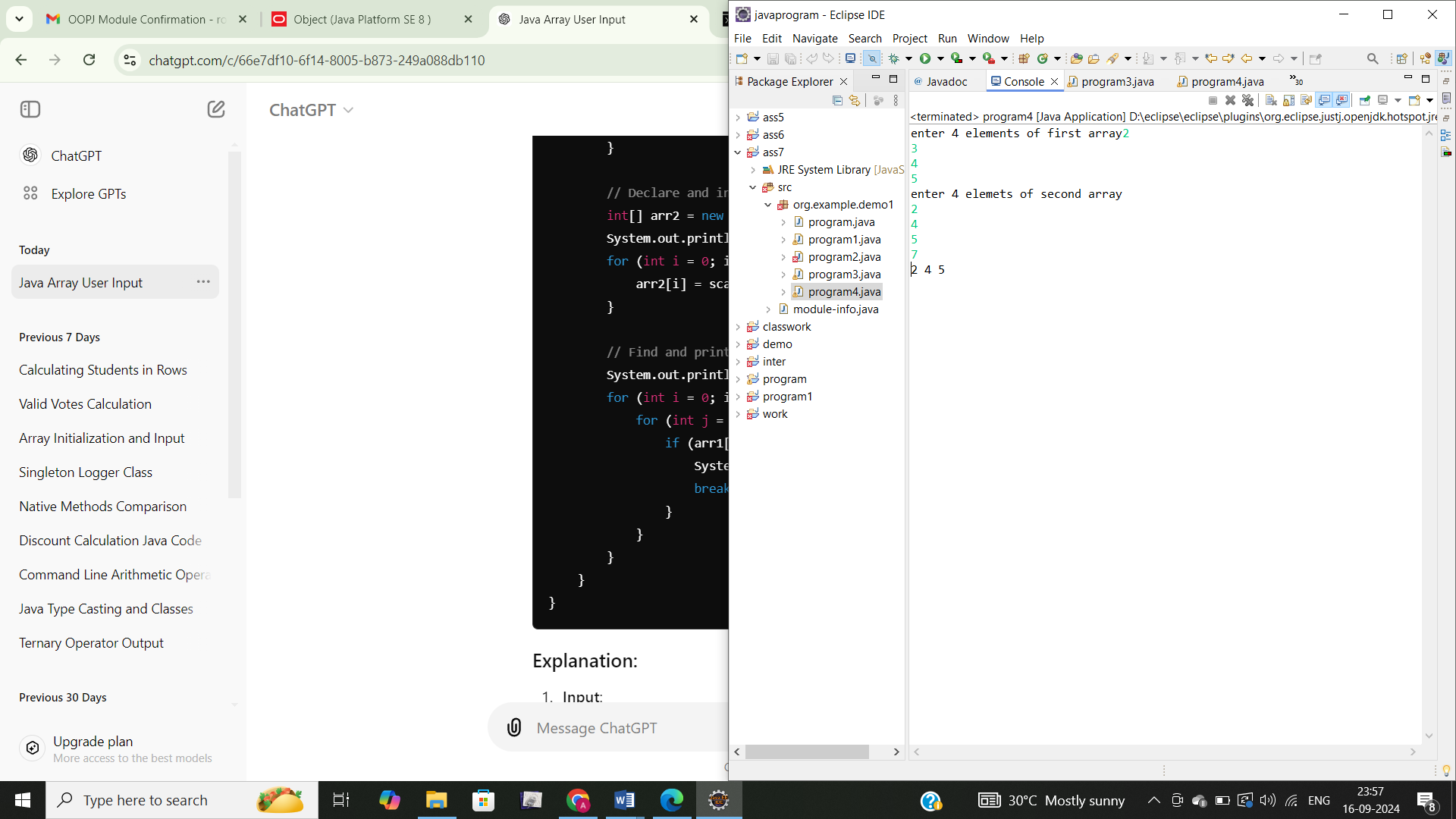

80. }
81. 
82. Write a program to remove duplicate elements from a single-dimensional array of
83. **package** org.example.demo1;
84. **import** java.util.Scanner;
85. **public** **class** program3 {
86. **public** **static** **int**[] removeDuplicate(**int**[]arr) {
87. **int**[]temp=**new** **int**[5];
88. **int** j=0;
89. **for**(**int** i=0;i<arr.length;i++) {
90. **boolean** isDuplicate = **false**;
91. **for**(**int** k=0;k<j;k++) {
92. **if**(arr[i]==temp[k]) {
93. isDuplicate=**true**;
94. **break**;
95. }
96. }
97. **if**(!isDuplicate) {
98. temp[j++]=arr[i];
99. }
101. }
102. **int**[]resultArray=**new** **int**[j];
103. System.*arraycopy*(temp,0,resultArray,0,j);
104. **return** resultArray;
105. }
107. **public** **static** **void** main(String[] args) {
108. Scanner sc=**new** Scanner(System.***in***);
110. **int**[]arr=**new** **int**[5];
111. System.***out***.println("enter 5 element");
112. **for**(**int** i=0;i<arr.length;i++) {
113. System.***out***.println("element"+(i+1)+":");
114. arr[i]=sc.nextInt();
116. }
117. **int**[] resultArray =*removeDuplicate*(arr);
118. **for**(**int** i= 0;i< resultArray.length;i++) {
120. System.***out***.print(resultArray[i]+" ");
121. }
122. }



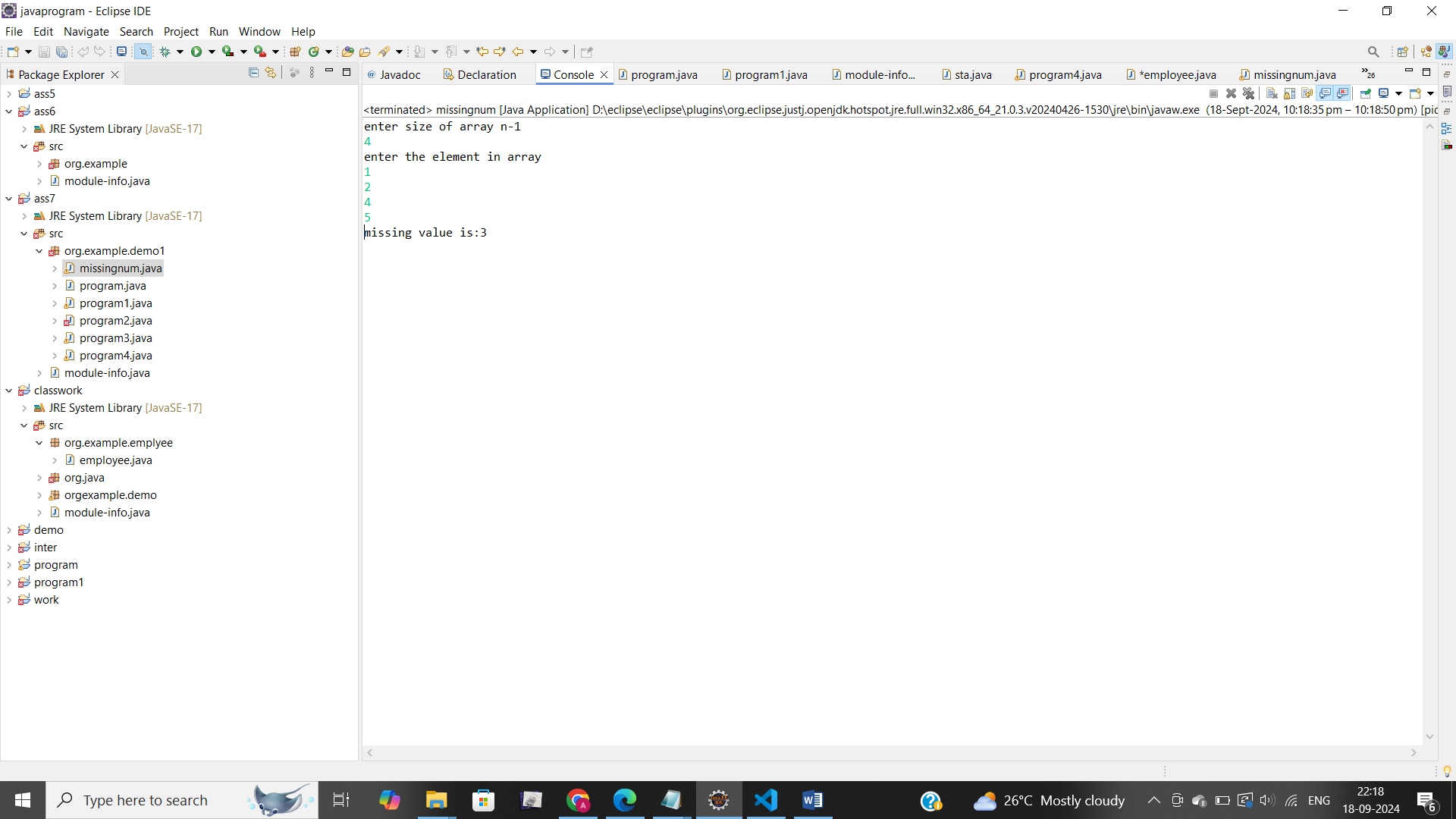
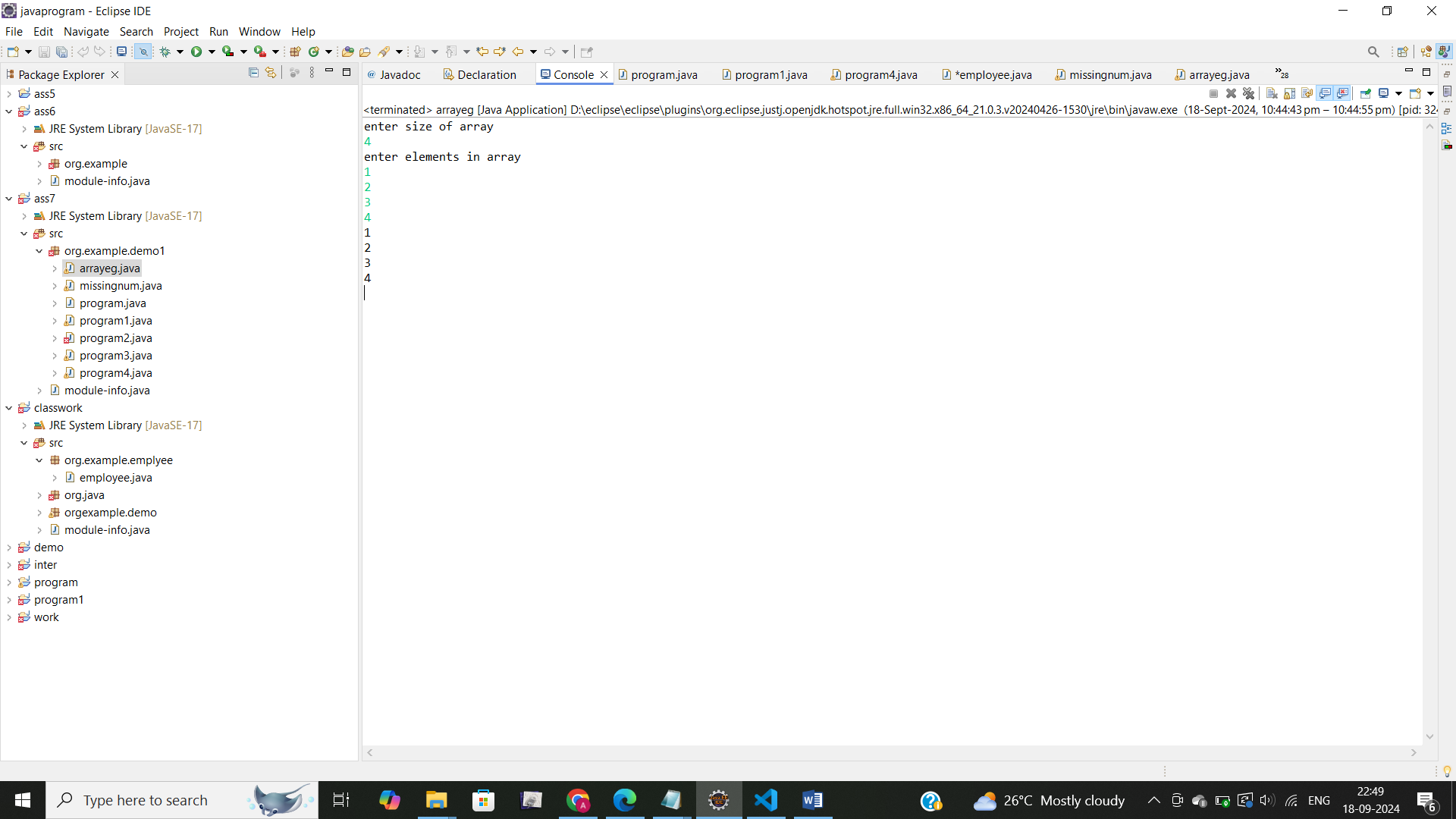
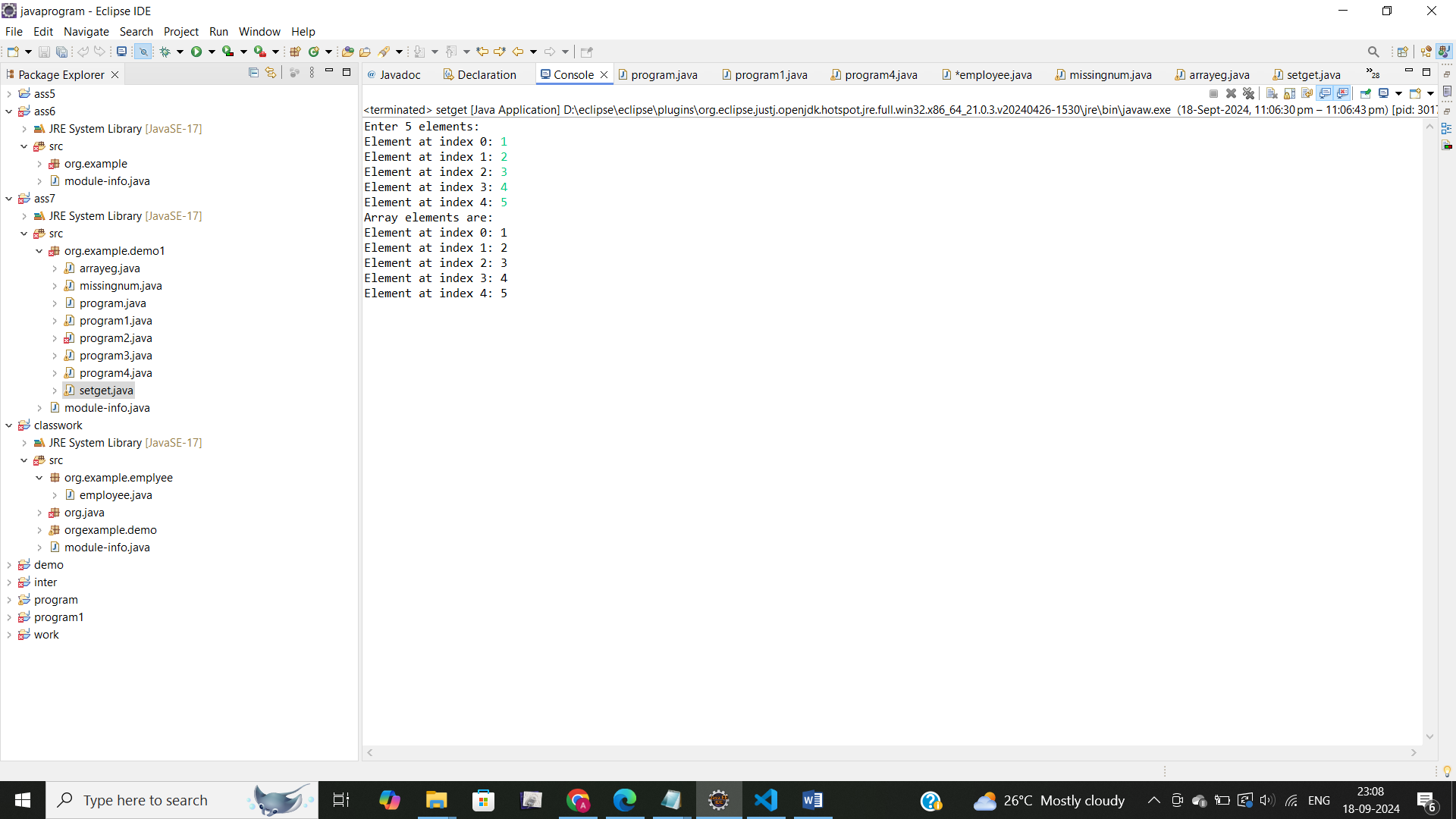

128. }
129. integers. 
130. Write a program to find the intersection of two single-dimensional arrays.
131. **package** org.example.demo1;
132. **import** java.util.Scanner;
133. **public** **class** program4 {
134. **public** **static** **void** main(String[] args) {
135. Scanner sc=**new** Scanner(System.***in***);
136. **int**[]arr1=**new** **int**[4];
137. System.***out***.print("enter 4 elements of first array");
138. **for**(**int** i=0;i<arr1.length;i++) {
140. arr1[i]=sc.nextInt();

143. }
144. **int**[]arr2=**new** **int**[4];
145. System.***out***.println("enter 4 elemets of second array" );
146. **for**(**int** i=0;i<arr2.length;i++) {
148. arr2[i]=sc.nextInt();

151. }
152. **for**(**int** i=0;i<4;i++) {
153. **for**(**int** j=0;j<4;j++) {
154. **if**(arr1[i]==arr2[j]) {
155. System.***out***.print(arr1[i]+" ");
156. **break**;
157. }
159. }


163. }
164. }
165. }
166. 
167. Write a program to find the missing number in an array of integers ranging from 1 to N.
168. **package** org.example.demo1;
169. **import** java.util.Scanner;
170. **public** **class** missingnum {
172. **public** **static** **void** main(String[] args) {
173. Scanner sc=**new** Scanner(System.***in***);
174. System.***out***.println("enter size of array n-1");
175. **int** n=sc.nextInt();
176. **int**[]arr=**new** **int**[n];
177. System.***out***.println("enter the element in array");
178. **for**(**int** i=0;i<n;i++) {
179. arr[i]=sc.nextInt();

182. }
183. **int** N=n+1;
184. **int** expectedsum=N\*(N+1)/2;
185. **int** actualsum=0;
186. **for**(**int** i=0;i<n;i++) {
187. actualsum +=arr[i];
189. }
190. **int** missingvalue=expectedsum-actualsum;
191. System.***out***.println("missing value is:"+missingvalue);

194. }
195. }
196. 
197. Declare a single-dimensional array as a field inside a class and instantiate it inside the class constructor. Define methods named acceptRecord and printRecord within the class and test their functionality.
198. **package** org.example.demo1;
199. **import** java.util.Scanner;
200. **public** **class** arrayeg {
201. **int**[]arr;
202. arrayeg(**int** size){
203. arr=**new** **int**[size];
205. }
206. **public** **void** acceptRecord() {
207. Scanner sc=**new** Scanner(System.***in***);
208. System.***out***.println("enter elements in array");
209. **for**(**int** i=0;i<arr.length;i++) {
210. arr[i]=sc.nextInt();
211. }
213. }
214. **public** **void** printRecord() {
215. **for**(**int** value:arr) {
216. System.***out***.println(value+" ");
217. }
219. }
220. **public** **static** **void** main(String[] args) {
221. Scanner sc=**new** Scanner(System.***in***);
222. System.***out***.println("enter size of array");
223. **int** size=sc.nextInt();
224. arrayeg a=**new** arrayeg(size);
225. a.acceptRecord();
226. a.printRecord();
228. }
229. }
230. 
231. Modify the previous assignment to use getter and setter methods instead of acceptRecord and printRecord.
232. **package** org.example.demo1;
233. **import** java.util.Scanner;
234. **public** **class** setget {
236. **private** **int**[] arr;
238. **public** setget(**int** size) {
239. arr = **new** **int**[size];
240. }
242. **public** **void** setRecord() {
243. Scanner scanner = **new** Scanner(System.***in***);
244. System.***out***.println("Enter " + arr.length + " elements:");
245. **for** (**int** i = 0; i < arr.length; i++) {
246. System.***out***.print("Element at index " + i + ": ");
247. arr[i] = scanner.nextInt();
248. }
249. }
251. **public** **void** getRecord() {
252. System.***out***.println("Array elements are:");
253. **for** (**int** i = 0; i < arr.length; i++) {
254. System.***out***.println("Element at index " + i + ": " + arr[i]);
255. }
256. }
258. **public** **static** **void** main(String[] args) {
260. setget example = **new** setget (5);
262. example.setRecord();
264. example.getRecord();
265. }
266. }
268. 
269. You need to implement a system to manage airplane seat assignments. The airplane has seats arranged in rows and columns. Implement functionalities to

* Initialize the seating arrangement with a given number of rows and columns.
* Book a seat to mark it as occupied.
* Cancel a booking to mark a seat as available.
* Check seat availability to determine if a specific seat is available.
* Display the current seating chart.