

```

1  /*****
2  *                               1                               *
3  *****/
4  #include <bits/stdc++.h>
5  using namespace std;
6
7  int main()
8  {
9      double allX[] = {1, 2, 3, 4, 5};
10     double allY[] = {1, 8, 27, 64, 125};
11     int tableSize = sizeof allY / sizeof allY[0];
12     vector<double> dely[tableSize];
13     double x = 1;
14     double h = allX[1] - allX[0];
15     double u = (x - allX[0]) / h;
16
17     for (int i = 0; i < tableSize; i++) {
18         dely[i].push_back(allY[i]);
19     }
20
21     for (int i = 1; i < tableSize; i++) {
22         for (int j = 1; j < dely[i - 1].size(); j++) {
23             dely[i].push_back(dely[i - 1][j] - dely[i - 1][j - 1]);
24         }
25     }
26
27     double firstD = dely[1][0] + (2 * u - 1) * (dely[2][0] / 2) + (3
* u * u - 6 * u + 2) * (dely[3][0] / 6) + (4 * u * u * u - 18 * u * u + 22
* u - 6) * (dely[4][0] / 24);
28     firstD /= h;
29
30     double secondD = dely[2][0] + (6 * u - 6) * (dely[3][0] / 6) + (
12 * u * u - 36 * u + 22) * (dely[4][0] / 24);
31     secondD /= (h * h);
32
33     cout << "First Derivative = " << setprecision(3) << fixed <<
firstD << "\n";
34     cout << "Second Derivative = " << setprecision(3) << fixed <<
secondD << "\n";
35
36     return 0;
37 }
38
39 /*****
40 *                               2                               *
41 *****/
42 #include <bits/stdc++.h>
43 using namespace std;
44
45 int main()
46 {
47     double allX[] = {1, 2, 3, 4, 5};
48     double allY[] = {1, 8, 27, 64, 125};
49     int tableSize = sizeof allY / sizeof allY[0];
50     vector<double> dely[tableSize];
51     double x = 1.5;
52     double h = allX[1] - allX[0];
53     double u = (x - allX[0]) / h;
54
55     for (int i = 0; i < tableSize; i++) {
56         dely[i].push_back(allY[i]);
57     }
58
59     for (int i = 1; i < tableSize; i++) {
60         for (int j = 1; j < dely[i - 1].size(); j++) {
61             dely[i].push_back(dely[i - 1][j] - dely[i - 1][j - 1]);

```

```

62     }
63 }
64
65     double firstD = dely[1][0] + (2 * u - 1) * (dely[2][0] / 2) + (3
* u * u - 6 * u + 2) * (dely[3][0] / 6) + (4 * u * u * u - 18 * u * u + 22
* u - 6) * (dely[4][0] / 24);
66     firstD /= h;
67
68     double secondD = dely[2][0] + (6 * u - 6) * (dely[3][0] / 6) + (
12 * u * u - 36 * u + 22) * (dely[4][0] / 24);
69     secondD /= (h * h);
70
71     cout << "First Derivative = " << setprecision(3) << fixed <<
firstD << "\n";
72     cout << "Second Derivative = " << setprecision(3) << fixed <<
secondD << "\n";
73
74     return 0;
75 }
76
77 /*****
78 *                               3                               *
79 *****/
80 #include <bits/stdc++.h>
81 using namespace std;
82
83 double fX(double x)
84 {
85     return (5 * log10(x));
86 }
87
88 int main()
89 {
90     double upperLimit = 7;
91     double lowerLimit = 1;
92     int n = 6;
93     double h = (upperLimit - lowerLimit) / n;
94
95     double x = lowerLimit;
96     double y[n + 1];
97     for (int i = 0; i <= n; i++) {
98         y[i] = fX(x);
99         x += h;
100     }
101
102     double I = y[0] + y[n];
103     for (int i = 1; i < n; i++) {
104         I += (2 * y[i]);
105     }
106     I = I * (h / 2);
107
108     cout << "Area = " << setprecision(3) << fixed << I << "\n";
109
110     return 0;
111 }
112
113 /*****
114 *                               4                               *
115 *****/
116 #include <bits/stdc++.h>
117 using namespace std;
118
119 #define PI acos(-1.0)
120 #define sinD(degree) sin((degree * PI) / 180.0)
121
122 double fX(double x)

```

```

123 {
124     return (PI * exp(sinD(x))) / 2;
125 }
126
127 int main()
128 {
129     double upperLimit = 7;
130     double lowerLimit = 1;
131     int n = 6;
132     double h = (upperLimit - lowerLimit) / n;
133
134     double x = lowerLimit;
135     double y[n + 1];
136     for (int i = 0; i <= n; i++) {
137         y[i] = fX(x);
138         x += h;
139     }
140
141     double I = y[0] + y[n];
142     for (int i = 1; i < n; i++) {
143         if (i % 2 == 0) {
144             I += (2 * y[i]);
145         }
146         else {
147             I += (4 * y[i]);
148         }
149     }
150     I = I * (h / 3);
151
152     cout << "Area = " << setprecision(3) << fixed << I << "\n";
153
154     return 0;
155 }
156
157 /*****
158  *                               5                               *
159  *****/
160 #include <bits/stdc++.h>
161 using namespace std;
162
163 double fX(double x)
164 {
165     return (x / (1 + x * x));
166 }
167
168 int main()
169 {
170     double upperLimit = 7;
171     double lowerLimit = 1;
172     int n = 6;
173     double h = (upperLimit - lowerLimit) / n;
174
175     double x = lowerLimit;
176     double y[n + 1];
177     for (int i = 0; i <= n; i++) {
178         y[i] = fX(x);
179         x += h;
180     }
181
182     double I = y[0] + y[n];
183     for (int i = 1; i < n; i++) {
184         if (i % 3 == 0) {
185             I += (2 * y[i]);
186         }
187         else {
188             I += (3 * y[i]);

```

```

189     }
190 }
191 I = I * ((3 * h) / 8);
192
193 cout << "Area = " << setprecision(3) << fixed << I << "\n";
194
195 return 0;
196 }
197
198 /*****
199 *                               6                               *
200 *****/
201 #include <bits/stdc++.h>
202 using namespace std;
203
204 int temp[4][4];
205
206 int det3x3()
207 {
208     int ans = 0;
209     ans += temp[1][1] * (temp[2][2] * temp[3][3] - temp[2][3] * temp[
3][2]);
210     ans -= temp[1][2] * (temp[2][1] * temp[3][3] - temp[2][3] * temp[
3][1]);
211     ans += temp[1][3] * (temp[2][1] * temp[3][2] - temp[2][2] * temp[
3][1]);
212     return ans;
213 }
214
215 int main()
216 {
217     int n = 4;
218     int matrix[n + 1][n + 1];
219
220     for (int i = 1; i <= n; i++) {
221         for (int j = 1; j <= n; j++) {
222             cin >> matrix[i][j];
223         }
224     }
225
226     int ans = 0;
227     for (int i = 1; i <= n; i++) {
228         for (int j = 2; j <= n; j++) {
229             int m = 1;
230             for (int k = 1; k <= n; k++) {
231                 if (k == i) {
232                     continue;
233                 }
234                 temp[j - 1][m] = matrix[j][k];
235                 m++;
236             }
237         }
238
239         cout << det3x3() << "\n";
240
241         int d = matrix[1][i] * det3x3();
242         if (i % 2 == 1) {
243             ans += d;
244         }
245         else {
246             ans -= d;
247         }
248     }
249
250     cout << "Determinant = " << ans << "\n";
251

```

```

252     return 0;
253 }
254
255 /*****
256 *                               7                               *
257 *****/
258 #include <bits/stdc++.h>
259 using namespace std;
260
261 const int n = 3;
262
263 double A[n + 1][n + 1] = {
264     {0, 0, 0, 0},
265     {0, 1, 1, 1},
266     {0, 1, 2, 3},
267     {0, 1, 3, 4}
268 };
269
270 double B[n + 1] = {0, 1, 6, 6};
271
272 double det3x3()
273 {
274     double ans = 0;
275     ans += A[1][1] * (A[2][2] * A[3][3] - A[2][3] * A[3][2]);
276     ans -= A[1][2] * (A[2][1] * A[3][3] - A[2][3] * A[3][1]);
277     ans += A[1][3] * (A[2][1] * A[3][2] - A[2][2] * A[3][1]);
278     return ans;
279 }
280
281 int main()
282 {
283     double detA = det3x3();
284     double cofactor[n + 1][n + 1];
285
286     vector<double> data;
287
288     data.clear();
289     for (int i = 1; i <= n; i++) {
290         for (int j = 1; j <= n; j++) {
291             for (int row = 1; row <= n; row++) {
292                 for (int col = 1; col <= n; col++) {
293                     if (row == i || col == j) {
294                         continue;
295                     }
296                     data.push_back(A[row][col]);
297                 }
298             }
299             cofactor[i][j] = (data[0] * data[3]) - (data[1] * data[2]);
300             if ((i + j) % 2 == 1) {
301                 cofactor[i][j] *= -1;
302             }
303             data.clear();
304         }
305     }
306
307     double adjA[n + 1][n + 1];
308     for (int i = 1; i <= n; i++) {
309         for (int j = 1; j <= n; j++) {
310             adjA[i][j] = cofactor[j][i];
311             adjA[i][j] /= detA;
312         }
313     }
314
315     double solution[n + 1];
316

```

```

317     for (int i = 1; i <= n; i++) {
318         double x = 0;
319         for (int j = 1; j <= n; j++) {
320             x += adjA[i][j] * B[j];
321         }
322         solution[i] = x;
323     }
324
325     cout << "x = " << setprecision(3) << fixed << solution[1] << "\n"
;
326     cout << "y = " << setprecision(3) << fixed << solution[2] << "\n"
;
327     cout << "z = " << setprecision(3) << fixed << solution[3] << "\n"
;
328
329     return 0;
330 }
331
332 /*****
333  *                               11                               *
334  *****/
335 #include <bits/stdc++.h>
336 using namespace std;
337
338 double eqX(double y, double z)
339 {
340     double x = (95 - 11 * y + 4 * z) / 83;
341     return x;
342 }
343
344 double eqY(double x, double z)
345 {
346     double y = (71 - 3 * x - 29 * z) / 8;
347     return y;
348 }
349
350 double eqZ(double x, double y)
351 {
352     double z = (104 - 7 * x - 52 * y) / 13;
353     return z;
354 }
355
356 int main()
357 {
358     double x, y, z;
359     int n = 5;
360
361     x = y = z = 0;
362
363     for (int i = 1; i <= n; i++) {
364         double newX = eqX(y, z);
365         double newY = eqY(x, z);
366         double newZ = eqZ(x, y);
367
368         x = newX;
369         y = newY;
370         z = newZ;
371     }
372
373     cout << "x = " << setprecision(3) << fixed << x << "\n";
374     cout << "y = " << setprecision(3) << fixed << y << "\n";
375     cout << "z = " << setprecision(3) << fixed << z << "\n";
376
377     return 0;
378 }
379

```

```

380  /*****
381  *                               12                               *
382  *****/
383  #include <bits/stdc++.h>
384  using namespace std;
385
386  double eqX(double y, double z)
387  {
388      double x = (12 - y - z) / 10;
389      return x;
390  }
391
392  double eqY(double x, double z)
393  {
394      double y = (13 - 2 * x - z) / 10;
395      return y;
396  }
397
398  double eqZ(double x, double y)
399  {
400      double z = (14 - 2 * x - 2 * y) / 10;
401      return z;
402  }
403
404  int main()
405  {
406      double x, y, z;
407      int n = 1000;
408
409      x = y = z = 0;
410
411      for (int i = 1; i <= n; i++) {
412          x = eqX(y, z);
413          y = eqY(x, z);
414          z = eqZ(x, y);
415      }
416
417      cout << "x = " << setprecision(3) << fixed << x << "\n";
418      cout << "y = " << setprecision(3) << fixed << y << "\n";
419      cout << "z = " << setprecision(3) << fixed << z << "\n";
420
421      return 0;
422  }

```