

```
1: //      Sec. 5.1 of lecture
2: //      2D arrays dynamic
3: //      Multiplication Matrix*Vector  f := A * u
4: //      with A as 2D array  or  B as 1D array
5: //  g++ -std=c++11 -Wall -Wextra -pedantic bsp514_b.cpp
6: #include <iostream>
7: #include <vector>
8: #include <cassert>                                // assert
9: using namespace std;
10:
11: int main()
12: {
13:     const int NROW = 4, MCOL = 3;
14:     vector<double> f(NROW);
15:     const vector<double> u({1, 2, 3});           // initialize u
16:     assert( MCOL == static_cast<int>(u.size()) );
17:
18: //-----
19: //      matrix as 2D-Array
20:     const vector<vector<double>> a({ {4, -1, -0.5}, { -1, 4, -1}, { -0.5, -1, 4}, {3, 0, -1} });
21:     assert( NROW == static_cast<int>(a.size()) );           // check number of rows
22:     assert( MCOL == static_cast<int>(a.back().size()) );     // check number of columns in last row
23:
24: //      matrix times vector : f := A * u
25:     for (int i = 0; i < NROW; ++i) {
26:         f[i] = 0.0;                                           // initialize f[i]
27:         for (int j = 0; j < MCOL; ++j) {
28:             f[i] = f[i] + a[i][j] * u[j];
29:         }
30:         cout << a[i].data() << endl;                          // Address of a[i][0]
31:     }
32: //      output result
33:     cout << endl;
34:     for (int i = 0; i < NROW; ++i) {
35:         cout << " " << f[i];
36:     }
37:     cout << endl << endl;
38:
39: //-----
40: //      matrix as 1D array with index calculation
41:     const vector<double> b({4, -1, -0.5, -1, 4, -1, -0.5, -1, 4, 3, 0, -1 });
42:     assert( NROW * MCOL == static_cast<int>(b.size()) );
43:
44: //      matrix times vector : f := B * u
45:     for (int i = 0; i < NROW; ++i) {
46:         f[i] = 0.0;                                           // initialize f[i]
47:         for (int j = 0; j < MCOL; ++j) {
48:             f[i] = f[i] + b[i * MCOL + j] * u[j];
49:         }
50:     }
51: //      output result
52:     cout << endl;
53:     for (int i = 0; i < NROW; ++i) {
54:         cout << " " << f[i];
55:     }
56:     cout << endl << endl;
57:
58:     return 0;
59: }
60:
61:
62:
63:
64:
65:
66:
67:
68:
```