Лабораторная работа номер 3
Вариант 5

Сделал: Каравка Денис Иванович
Группа: 1БПМИ-УДМО

Постановка задачи:

Даны 6-элементные вещественные векторы x и y и квадратные матрицы A, B и C 6-го порядка. Вычислить величину (Ax, By)+(Cx,y)/(x, By).

Исходные данные:

x, y – шестиэлементный массив вещественных чисел

size – размерность вектора и порядок матрицы

A, C, B – шестиэлементный двумерный массив вещественных чисел

Результаты выполнения алгоритма:

scalarResultAxBy – величина (Ax, By)+(Cx,y)/(x, By).

Математическая формула условия задачи

1) resultFirst =$\sum_{i=0}^{size}(\sum_{j=0}^{size} Aij * Xi)$
2) resultSecond =$\sum_{i=0}^{size}(\sum_{j=0}^{size} Bij * Yi)$
3) resultThird = $\sum_{i=0}^{size}(\sum_{j=0}^{size} Cij * Xi)$
4) scalarResultAxBy =$\sum_{i=0}^{size}(Axi * Byi)$
5) scalarResultCxY = $\sum_{i=0}^{size}(Cxi * Yi)$
6) scalarResultXBy = $\sum_{i=0}^{size}(Xi * Byi)$
7) scalarResultAxBy = scalarResultAxBy + scalarResultCxY / scalarResultXBy.

Таблица разработки программы

№ этапа	Этапы разработки	Примечание

1	/*Занесения значений в векторы*/
	void inputValuesIntoVector(int size, long vector[size]){
	  cout << "Ввод значений вектора:" << endl;
	}
	/*Занесения значений в матрицу*/
	template <typename T, size_t size>
	void inputValuesIntoMatrix(T (&matrix)[size][size]){
	cout << "Ввод значений матрицы:" << endl;
	}
	/*Умножение матрицы и вектора*/
	template <typename T, size_t size>
	long* multipleMatrixAndVector(T (&matrix)[size][size],
	long vector[size], long resultArray[size]){
	  cout << "Умножение матрицы и вектора" << endl;
	}
	/*Скалярное произведение*/
	float scalarMultiple(int size, long resultVector[size],
	long vector[size]){
	  int result = 2;
	  cout << "Скалярное произведение равно: " << result <<
	endl;
	  return result;
	}
	template <typename T, size_t size>
	void outputMatrix(T (&matrix)[size][size]){
	  cout << "Вывод матрицы" << endl;
	}
	void outputVector(int size, long vector[size]){
	  cout << "Вывод вектора" << endl;
	}
	/*Главная функция*/
	int main()
	{
	  const int size = 6;
	  long vectorX[size];
	  long vectorY[size];
	  float matrixA[size][size];
	  float matrixB[size][size];
	  float matrixC[size][size];
	  long resultFirst[size];
	  long resultSecond[size];
	  long resultThird[size];
	  float scalarResultAxBy = 0;
	  float scalarResultCxY = 0;
	  float scalarResultXBy = 0;
		long vectorX, vectorY;
		
		float matrixA, matrixB, matrixC;
		
		long resultFirst, resultSecond,
		resultThird;
		
		float scalarResultAxBy,
		scalarResultCxY,
		scalarResultXBy;
		
		float matrixA, matrixB, matrixC;
		
		long vectorX, vectorY,
		resultFirst, resultSecond,
		resultThird;

```
  inputValuesIntoMatrix(matrixA);
  inputValuesIntoMatrix(matrixB);
  inputValuesIntoMatrix(matrixC);
  inputValuesIntoVector(size, vectorX);
  inputValuesIntoVector(size, vectorY);
  multipleMatrixAndVector(matrixA, vectorX, resultFirst);
  multipleMatrixAndVector(matrixB, vectorY,
resultSecond);
  multipleMatrixAndVector(matrixC, vectorX, resultThird);
  scalarResultAxBx = scalarMultiple(size, resultFirst,
resultSecond);
  scalarResultXBy = scalarMultiple(size, vectorX,
resultSecond);
  scalarResultCxY = scalarMultiple(size, resultThird,
vectorY);

  cout << "(Ax, By)+(Cx,y) = ";
  cout << (scalarResultAxBx+scalarResultCxY) << endl;
  cout << "(Ax, By)+(Cx,y)/(x, By) = ";
  cout <<
((scalarResultAxBx+scalarResultCxY)/scalarResultXBy) <<
endl;

}
```

2

```
#include <iostream>
#include <math.h>
using namespace std;
/*
Даны 6-элементные вещественные векторы x и y и квадратные
матрицы A, B и C 6-го порядка. Вычислить величину (Ax,
By)+(Cx,y)/(x, By).
*/

/*Занесение значений в векторы*/
void inputValuesIntoVector(int size, long vector[size]){
  for(int i = 0; i < size; i++){
    vector[i] = 2;
  }
}

/*Занесение значений в матрицу*/
template <typename T, size_t size>
void inputValuesIntoMatrix(T (&matrix)[size][size]){
  int j = 0;
  for(int i = 0; i < size && j < size; i++){
    matrix[j][i] = i;
    if(i+1 == size){
      j++;
      i = -1;
    }
```

```
  }
}

/*Умножение матрицы и вектора*/
template <typename T, size_t size>
long* multipleMatrixAndVector(T (&matrix)[size][size],
long vector[size], long resultArray[size]){
  int j = 0;
  int summaryMatrix = 0;
  int summaryVector = 0;
  for(int i = 0; i < size && j < size; i++){
    summaryMatrix += matrix[j][i]*vector[i];
    if(i+1 == size){
      resultArray[j] = summaryMatrix;
      summaryMatrix = 0;
      summaryVector = 0;
      j++;
      i = -1;
    }
  }
  return resultArray;
}

/*Скалярное произведение*/
float scalarMultiple(int size, long resultVector[size],
long vector[size]){
  float result = 0;
  for(int i = 0; i < size; i++){
    result += (resultVector[i] * vector[i]);
  }
  return result;
}

/*Вывод матрицы*/
template <typename T, size_t size>
void outputMatrix(T (&matrix)[size][size]){
  int j = 0;
  for(int i = 0; i < size && j < size; i++){
    cout << matrix[j][i] << " ";
    if(i+1 == size){
      cout << endl;
      j++;
      i = -1;
    }
  }
}

/*Вывод вектора*/
void outputVector(int size, long vector[size]){
  for(int i = 0; i < size; i++){
```

```
    cout << vector[i] << " ";
  }
  cout << endl;
}
/*Главная функция*/
int main()
{
  const int size = 6;
  long vectorX[size];
  long vectorY[size];
  float matrixA[size][size];
  float matrixB[size][size];
  float matrixC[size][size];
  long resultFirst[size];
  long resultSecond[size];
  long resultThird[size];
  float scalarResultAxBx = 0;
  float scalarResultCxY = 0;
  float scalarResultXBy = 0;

  cout << "Матрица A:" << endl;
  inputValuesIntoMatrix(matrixA);
  outputMatrix(matrixA);

  cout << "Матрица B:" << endl;
  inputValuesIntoMatrix(matrixB);
  outputMatrix(matrixB);

  cout << "Матрица C:" << endl;
  inputValuesIntoMatrix(matrixC);
  outputMatrix(matrixC);

  cout << "Вектор X:" << endl;
  inputValuesIntoVector(size, vectorX);
  outputVector(size, vectorX);

  cout << "Вектор Y:" << endl;
  inputValuesIntoVector(size, vectorY);
  outputVector(size, vectorY);

  cout << "Произведение Ax:" << endl;
  multipleMatrixAndVector(matrixA, vectorX, resultFirst);
  outputVector(size, resultFirst);

  cout << "Произведение By:" << endl;
  multipleMatrixAndVector(matrixB, vectorY,
resultSecond);
  outputVector(size, resultSecond);

  cout << "Произведение Cx:" << endl;
```

```
  multipleMatrixAndVector(matrixC, vectorX, resultThird);
  outputVector(size, resultThird);

  cout << "Скалярное произведение Ax,Bx: ";
  scalarResultAxBx = scalarMultiple(size, resultFirst,
resultSecond);
  cout << scalarResultAxBx << endl;

  cout << "Скалярное произведение X,By: ";
  scalarResultXBy = scalarMultiple(size, vectorX,
resultSecond);
  cout << scalarResultXBy << endl;

  cout << "Скалярное произведение Cx,Y: ";
  scalarResultCxY = scalarMultiple(size, resultThird,
vectorY);
  cout << scalarResultCxY << endl;

  cout << "(Ax, By)+(Cx,y) = ";
  cout << (scalarResultAxBx+scalarResultCxY) << endl;

  cout << "(Ax, By)+(Cx,y)/(x, By) = ";
  cout <<
((scalarResultAxBx+scalarResultCxY)/scalarResultXBy) <<
endl;
}
```

Текст программы

Текст программы с «заглушками-функциями»:

```
#include <iostream>
#include <math.h>
using namespace std;
/*Занесения значений в векторы*/
void inputValuesIntoVector(int size, long vector[size]){
  cout << "Ввод значений вектора:" << endl;
}
/*Занесения значений в матрицу*/
template <typename T, size_t size>
void inputValuesIntoMatrix(T (&matrix)[size][size]){
   cout << "Ввод значений матрицы:" << endl;
}
/*Умножение матрицы и вектора*/
template <typename T, size_t size>
long* multipleMatrixAndVector(T (&matrix)[size][size], long vector[size], long resultArray[size]){
  cout << "Умножение матрицы и вектора" << endl;
}
/*Скалярное произведение*/
float scalarMultiple(int size, long resultVector[size], long vector[size]){
  int result = 2;
  cout << "Скалярное произведение равно: " << result << endl;
  return result;
}
template <typename T, size_t size>
void outputMatrix(T (&matrix)[size][size]){
  cout << "Вывод матрицы" << endl;
}
void outputVector(int size, long vector[size]){
  cout << "Вывод вектора" << endl;
}
/*Главная функция*/
int main()
{
  const int size = 6;
  long vectorX[size];
  long vectorY[size];
  float matrixA[size][size];
  float matrixB[size][size];
  float matrixC[size][size];
  long resultFirst[size];
  long resultSecond[size];
  long resultThird[size];
  float scalarResultAxBy = 0;
  float scalarResultCxY = 0;
  float scalarResultXBy = 0;
```

```
  inputValuesIntoMatrix(matrixA);
  inputValuesIntoMatrix(matrixB);
  inputValuesIntoMatrix(matrixC);
  inputValuesIntoVector(size, vectorX);
  inputValuesIntoVector(size, vectorY);
  multipleMatrixAndVector(matrixA, vectorX, resultFirst);
  multipleMatrixAndVector(matrixB, vectorY, resultSecond);
  multipleMatrixAndVector(matrixC, vectorX, resultThird);
  scalarResultAxBx = scalarMultiple(size, resultFirst, resultSecond);
  scalarResultXBy = scalarMultiple(size, vectorX, resultSecond);
  scalarResultCxY = scalarMultiple(size, resultThird, vectorY);
  cout << "(Ax, By)+(Cx,y) = ";
  cout << (scalarResultAxBx+scalarResultCxY) << endl;
  cout << "(Ax, By)+(Cx,y)/(x, By) = ";
  cout << ((scalarResultAxBx+scalarResultCxY)/scalarResultXBy) << endl;
}
```

Текст программы с реализованными процедурами и функциями:

```
#include <iostream>
#include <math.h>
using namespace std;
/*
Даны 6-элементные вещественные векторы x и y и квадратные матрицы A, B и C 6-го порядка. Вычислить
величину (Ax, By)+(Cx,y)/(x, By).
*/

/*Занесение значений в векторы*/
void inputValuesIntoVector(int size, long vector[size]){
  for(int i = 0; i < size; i++){
    cout << "Введите значение “ << i << ” элемента вектора: “ << endl;
    cin >> vector[i];
  }
}

/*Занесение значений в матрицу*/
template <typename T, size_t size>
void inputValuesIntoMatrix(T (&matrix)[size][size]){
  int j = 0;
  for(int i = 0; i < size && j < size; i++){
    cout << "Введите значение [“ << i << ”][” << j << ”] элемента матрицы: “ << endl;
    cin >> matrix[j][i];
    if(i+1 == size){
      j++;
      i = -1;
    }
  }
}

/*Умножение матрицы и вектора*/
template <typename T, size_t size>
```

```
long* multipleMatrixAndVector(T (&matrix)[size][size], long vector[size], long resultArray[size]){
  int j = 0;
  int summaryMatrix = 0;
  int summaryVector = 0;
  for(int i = 0; i < size && j < size; i++){
    summaryMatrix += matrix[j][i]*vector[i];
    if(i+1 == size){
      resultArray[j] = summaryMatrix;
      summaryMatrix = 0;
      summaryVector = 0;
      j++;
      i = -1;
    }
  }
  return resultArray;
}

/*Скалярное произведение*/
float scalarMultiple(int size, long resultVector[size], long vector[size]){
  float result = 0;
  for(int i = 0; i < size; i++){
    result += (resultVector[i] * vector[i]);
  }
  return result;
}

/*Вывод матрицы*/
template <typename T, size_t size>
void outputMatrix(T (&matrix)[size][size]){
  int j = 0;
  for(int i = 0; i < size && j < size; i++){
    cout << matrix[j][i] << " ";
    if(i+1 == size){
      cout << endl;
      j++;
      i = -1;
    }
  }
}

/*Вывод вектора*/
void outputVector(int size, long vector[size]){
  for(int i = 0; i < size; i++){
    cout << vector[i] << " ";
  }
  cout << endl;
}
/*Главная функция*/
int main()
{
```

```
const int size = 6;
long vectorX[size];
long vectorY[size];
float matrixA[size][size];
float matrixB[size][size];
float matrixC[size][size];
long resultFirst[size];
long resultSecond[size];
long resultThird[size];
float scalarResultAxBx = 0;
float scalarResultCxY = 0;
float scalarResultXBy = 0;

cout << "Матрица A:" << endl;
inputValuesIntoMatrix(matrixA);
outputMatrix(matrixA);

cout << "Матрица B:" << endl;
inputValuesIntoMatrix(matrixB);
outputMatrix(matrixB);

cout << "Матрица C:" << endl;
inputValuesIntoMatrix(matrixC);
outputMatrix(matrixC);

cout << "Вектор X:" << endl;
inputValuesIntoVector(size, vectorX);
outputVector(size, vectorX);

cout << "Вектор Y:" << endl;
inputValuesIntoVector(size, vectorY);
outputVector(size, vectorY);

cout << "Произведение Ax:" << endl;
multipleMatrixAndVector(matrixA, vectorX, resultFirst);
outputVector(size, resultFirst);

cout << "Произведение By:" << endl;
multipleMatrixAndVector(matrixB, vectorY, resultSecond);
outputVector(size, resultSecond);

cout << "Произведение Cx:" << endl;
multipleMatrixAndVector(matrixC, vectorX, resultThird);
outputVector(size, resultThird);

cout << "Скалярное произведение Ax,Bx: ";
scalarResultAxBx = scalarMultiple(size, resultFirst, resultSecond);
cout << scalarResultAxBx << endl;

cout << "Скалярное произведение X,By: ";
```

```
  scalarResultXBy = scalarMultiple(size, vectorX, resultSecond);
  cout << scalarResultXBy << endl;

  cout << "Скалярное произведение Cx,Y: ";
  scalarResultCxY = scalarMultiple(size, resultThird, vectorY);
  cout << scalarResultCxY << endl;
  if(scalarResultXBy != 0){
    cout << "(Cx,y)/(x, By) = ";
    cout << (scalarResultCxY/scalarResultXBy) << endl;
  }
  else{
    cout << "(x, By) = 0, ошибка деления на 0 \nВыход из программы";
    return 0;
  }
  cout << "(Ax, By)+(Cx,y)/(x, By) = ";
  cout << (scalarResultAxBx+scalarResultCxY/scalarResultXBy) << endl;
}
```

Тестирование программы

Тестовые данные номер 1:

vectorX = {2,2,2,2,2,2}

vectorY = {2,2,2,2,2,2}

matrixA = {
{0,1,2,3,4,5},
{0,1,2,3,4,5},
{0,1,2,3,4,5},
{0,1,2,3,4,5},
{0,1,2,3,4,5},
{0,1,2,3,4,5}}

matrixB = {
{0,1,2,3,4,5},
{0,1,2,3,4,5},
{0,1,2,3,4,5},
{0,1,2,3,4,5},
{0,1,2,3,4,5},
{0,1,2,3,4,5}}

matrixC = {
{0,1,2,3,4,5},
{0,1,2,3,4,5},
{0,1,2,3,4,5},
{0,1,2,3,4,5},
{0,1,2,3,4,5},
{0,1,2,3,4,5}}

Ожидаемый результат:

scalarResultCxY = 360;
scalarResultXBy = 360;
scalarResultAxBy = 5400;
scalarResultAxBy = scalarResultAxBy + scalarResultXBy / scalarResultCxY = 5400 + 360 / 360 = 5401;