Lab 6

Source Code

// Preston Knibbe

// October 22, 2015

#include <iostream>

#include <stdlib.h>

#include <time.h>

using namespace std;

const int LENGTH = 10;

void randSeed();

int randomNum(int, int);

void fillArray(int [][LENGTH]);

void printArray(const int [][LENGTH]);

int sumTotal(const int [][LENGTH]);

int minNum(const int [][LENGTH]);

int maxNum(const int [][LENGTH]);

int arrayAvg(const int [][LENGTH]);

int findColSum(int, const int [][LENGTH]);

int main(void) {

int nums[LENGTH][LENGTH];

randSeed();

fillArray(nums);

printArray(nums);

cout << "Sum: " << sumTotal(nums) << endl;

cout << "Min: " << minNum(nums) << endl;

cout << "Max: " << maxNum(nums) << endl;

cout << "Average: " << arrayAvg(nums) << endl;

cout << "Sum of column 3: " << findColSum(3,nums) << endl;

return 0;

}

// Seeds random number generator

void randSeed() {

srand(time(NULL));

}

// Creates a random number between the minLimit and maxLimit provided

int randomNum(int minLimit, int maxLimit) {

int randNum = rand() % maxLimit + minLimit;

return randNum;

}

// Fills array with random numbers

void fillArray(int nums[][LENGTH]) {

for (int i = 0; i < LENGTH; i++) {

for (int x = 0; x < LENGTH; x++) {

nums[i][x] = randomNum(10,90);

}

}

}

void printArray(const int nums[][LENGTH]) {

for (int x = 0; x < LENGTH; x++) {

for (int i = 0; i < LENGTH; i++) {

cout << nums[x][i] << " ";

}

cout << endl;

}

}

// Finds total sum of array components

int sumTotal(const int nums[][LENGTH]) {

int totalSum = 0;

for (int a = 0; a < LENGTH; a++) {

for (int b = 0; b < LENGTH; b++) {

totalSum += nums[a][b];

}

}

return totalSum;

}

// Finds minimum number in array

int minNum(const int nums[][LENGTH]) {

int num = 100000000;

for (int a = 0; a < LENGTH; a++) {

for (int b = 0; b < LENGTH; b++) {

if (nums[a][b] < num) {

num = nums[a][b];

}

}

}

return num;

}

// Finds maximum number in array

int maxNum(const int nums[][LENGTH]) {

int num = 0;

for (int a = 0; a < LENGTH; a++) {

for (int b = 0; b < LENGTH; b++) {

if (nums[a][b] > num) {

num = nums[a][b];

}

}

}

return num;

}

// Finds array average

int arrayAvg(const int nums[][LENGTH]) {

int sumAvg = 0;

int counter = 0;

for (int a = 0; a < LENGTH; a++) {

for (int b = 0; b < LENGTH; b++) {

sumAvg += nums[a][b];

counter++;

}

}

return (sumAvg/counter);

}

int findColSum(int col, const int nums[][LENGTH]) {

int colSum = 0;

for (int a = 0; a < LENGTH; a++) {

colSum += nums[col][a];

}

return colSum;

}

Code Output

50 37 41 13 46 98 89 79 25 56

10 17 39 89 84 33 95 19 72 67

70 93 70 18 43 53 50 16 57 89

91 38 80 35 60 60 88 96 85 22

41 93 13 94 91 29 27 15 94 25

33 75 46 11 59 73 33 41 70 52

41 76 94 43 34 64 60 65 84 93

40 32 80 21 13 67 48 99 60 22

78 34 78 68 30 66 11 34 67 94

50 61 23 47 37 21 97 65 19 22

Sum: 5426

Min: 10

Max: 99

Average: 54

Sum of column 3: 655

Process returned 0 (0x0) execution time : 0.021 s

Press any key to continue.

29 67 71 53 64 14 94 92 99 39

54 42 83 82 77 85 97 33 79 45

70 33 31 52 93 32 27 48 64 31

95 97 60 98 56 66 51 50 10 92

85 91 23 55 87 46 61 44 82 26

20 33 41 98 55 96 28 84 78 14

52 15 69 98 63 25 28 49 82 46

41 17 21 39 50 62 25 14 99 89

38 69 89 48 67 28 55 72 21 13

16 38 80 79 79 68 48 39 66 85

Sum: 5684

Min: 10

Max: 99

Average: 56

Sum of column 3: 675

Process returned 0 (0x0) execution time : 0.028 s

Press any key to continue.

21 63 73 14 15 58 31 97 44 89

47 64 99 86 75 83 59 21 29 47

85 50 14 49 40 47 83 94 82 48

42 88 38 45 44 83 39 90 64 82

60 35 13 25 16 29 23 97 34 21

33 53 96 36 32 16 99 40 53 28

25 25 81 43 63 18 25 84 16 60

52 53 52 45 63 21 71 80 47 39

33 91 95 55 84 90 10 30 38 79

32 83 93 13 74 85 40 55 91 43

Sum: 5338

Min: 10

Max: 99

Average: 53

Sum of column 3: 615

Process returned 0 (0x0) execution time : 0.019 s

Press any key to continue.