**CPS 450**

**Spring 2017**

**C Functions Lab**

**Due Mar. 24, 2017**

**Name \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

1. Consider the following program written in C syntax:

void swap (int a, int b) {

int temp;

temp = a;

a = b;

b = temp;

}

int main() {

int value = 2, list [5] = {1, 3, 5, 7, 9};

swap(value, list [0]);

swap (list[0], list [1]);

swap (value, list [value]);

return 0;

}

For each of the following parameter‐passing methods, what are all of the values of the variables

and list values after each of the three calls to swap?

a. Passed by value 2, 1 3 5 7 9

b. Passed by reference 2, 3 1 5 7 9

c. Passed by value‐result 2, 1 3 5 7 9

1. Consider the following program written in C syntax:

void fun (int first, int second) {

first+ = first;

second+=second;

}

int main() {

int list[2] = {1, 3};

fun (list[0], list[1]);

return 0;

}

For each of the following parameter‐passing methods, what are the values of the list array after execution?

a. Passed by value 1, 3

b. Passed by reference 2, 6

c. Passed by value‐result 1, 3

1. Write a program in C that has both static and stack-dynamic local variables in subprograms. Create six large (at least 1000x1000) matrices in the subprogram - three static and three stack-dynamic. Fill two of the static matrices and two of the stack-dynamic matrices with random numbers in the range 1-100, inclusive. The code in the subprogram must perform a large number of matrix operations on the static matrices and time the process, assigning the result to the third static matrix. Then it must repeat this with the stack-dynamic matrices. Compare and explain the results.

#include <ctime>

#include <cstdlib>

#include <iostream>

using namespace std;

void staticMatrix();

void stackDynamicMatrix();

int main() {

staticMatrix(); //Ignore this result

staticMatrix();

staticMatrix();

staticMatrix();

stackDynamicMatrix(); //Ignore this result

stackDynamicMatrix();

stackDynamicMatrix();

stackDynamicMatrix();

return 0;

}

void staticMatrix(){

static int a[1000][1000];

static int b[1000][1000];

static int c[1000][1000];

// Fill with random numbers

for (int i = 0; i < 100; i++)

for (int j = 0; j < 100; j++) {

a[i][j] = rand() % 100 + 1;

b[i][j] = rand() % 100 + 1;

}

size\_t Sum = 0

for (double k = 0; k < 300000; k++)

for (int i = 0; i < 100; i++)

for (int j = 0; j < 100; j++) {

c[i][j] = a[i][j] \* b[i][j];

Sum += c[i][j];

}

cout < <"Sum = " << Sum << endl;

return;

}

void stackDynamicMatrix(){

int d[1000][1000];

int e[1000][1000];

int f[1000][1000];

// Fill with random numbers

for (int i = 0; i < 100; i++)

for (int j = 0; j < 100; j++) {

d[i][j] = rand() % 100 + 1;

e[i][j] = rand() % 100 + 1;

}

size\_t Sum2 = 0;

for (double k = 0; k < 300000; k++)

for (int i = 0; i < 100; i++)

for (int j = 0; j < 100; j++) {

f[i][j] = d[i][j] \* e[i][j];

Sum += c[i][j];

}

cout < <"Sum = " << Sum2 << endl;

return;

}

Running your code unmodified (except for a more accurate timer) in MSVC 2010 get the following results in a release build:

Static = 2400 ms

Stack-Dynamic = 2400 ms