Fundamental Exercise on Computer and Information Engineering 1B Assignment 2

XL14608 Thiago Machado da Silva

April 30, 2015

I had a health problem and I missed last class.

Question 3

Question 2-1, 2-2 and 2-3

Referring to Figure 1, the left half shows the source codes, `prog2-1a.c` and `prog2-1b.c`, from left to right. The only difference between those is the pointer usage in lines 18-23 (`prog2-1b.c`) replacing lines 18-25 (`prog2-1a.c`). The right half of this figure shows compilation/execution and outputs, from top to bottom. The outputs have no difference, but in this case, the execution time of `prog2-1b` was about 5x faster than `prog2-1a`, since a pointer operation in the faster case was compensating 5000 integer operations in the slower one.

Question 3-1

Referring to Figure 2, the left half shows the source code of `prog2-2.c`. The solution is found by the Cramer's rule for a system of linear equations, explicit for a 2x2 input matrix and a 2x1 output vector. The right half of this figure shows compilation and output. The output's also explicit for the inputed matrix and vector sizes.

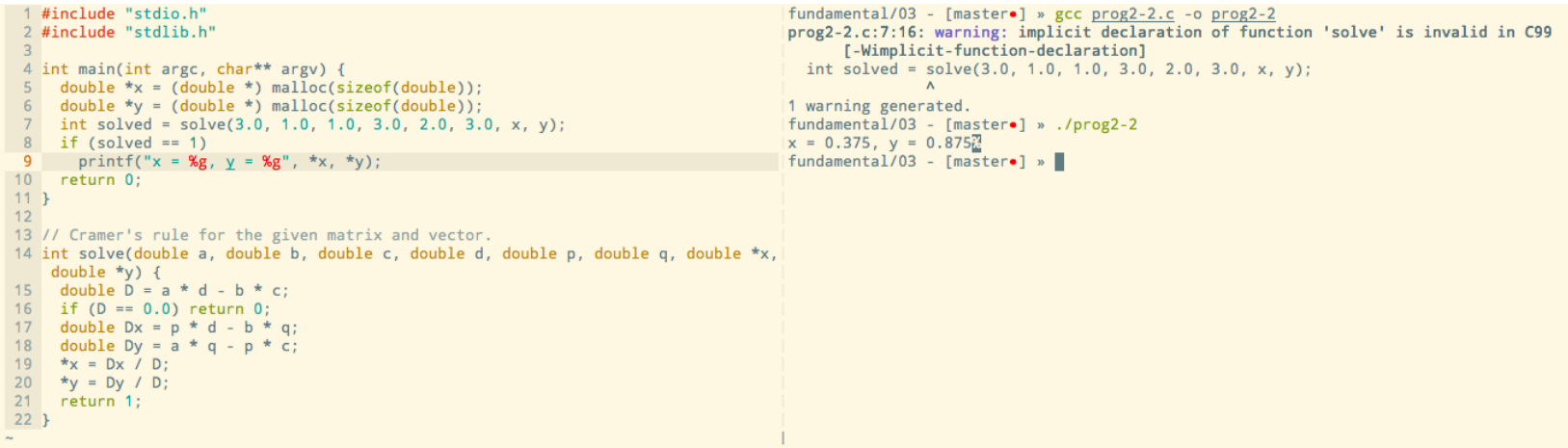
```c
#include "stdio.h"
#include "stdlib.h"

int main(int argc, char** argv) {

  int max = 5000, iter = 1000;
  int *array1 =
    (int*) malloc(sizeof(int) * max);
  int *array2 =
    (int*) malloc(sizeof(int) * max);

  // create arrays
  for (int i = 0; i < max; i++) {
    array1[i] = 2 * i + 1;
    array2[i] = 2 * i;
  }

  // exchange arrays elements
  for (int j = 0; j < iter; j++) {
    for (int i = 0; i < max; i++) {
      int k = array1[i];
      array1[i] = array2[i];
      array2[i] = k;
    }
  }

  // show arrays elements
  for (int i = 0; i < max; i++) {
    printf("%d, ", array1[i]);
    printf("%d\n", array2[i]);
  }

  return 0;
}
```

```c
#include "stdio.h"
#include "stdlib.h"

int main(int argc, char** argv) {

  int max = 5000, iter = 1000;
  int *array1 =
    (int*) malloc(sizeof(int) * max);
  int *array2 =
    (int*) malloc(sizeof(int) * max);

  // create arrays
  for (int i = 0; i < max; i++) {
    array1[i] = 2 * i + 1;
    array2[i] = 2 * i;
  }

  // exchange arrays addresses
  for (int j = 0; j < iter; j++) {
    int *arr = array1;
    array1 = array2;
    array2 = arr;
  }

  // show arrays elements
  for (int i = 0; i < max; i++) {
    printf("%d, ", array1[i]);
    printf("%d\n", array2[i]);
  }

  return 0;
}
```

```
fundamental/03 - [master•] » gcc prog2-1a.c -o prog2-1a
fundamental/03 - [master•] » gcc prog2-1b.c -o prog2-1b
fundamental/03 - [master•] » time ./prog2-1a > prog2-1a.out
./prog2-1a > prog2-1a.out  0.03s user 0.00s system 92% cpu 0.032 total
fundamental/03 - [master•] » time ./prog2-1b > prog2-1b.out
./prog2-1b > prog2-1b.out  0.00s user 0.00s system 68% cpu 0.007 total
fundamental/03 - [master•] » diff prog2-1a.out prog2-1b.out
fundamental/03 - [master•] »
```

```
1, 0                                    1, 0
3, 2                                    3, 2
5, 4                                    5, 4
7, 6                                    7, 6
9, 8                                    9, 8
11, 10                                  11, 10
13, 12                                  13, 12
15, 14                                  15, 14
17, 16                                  17, 16
19, 18                                  19, 18
21, 20                                  21, 20
23, 22                                  23, 22
25, 24                                  25, 24
27, 26                                  27, 26
29, 28                                  29, 28
31, 30                                  31, 30
33, 32                                  33, 32
35, 34                                  35, 34
37, 36                                  37, 36
39, 38                                  39, 38
41, 40                                  41, 40
43, 42                                  43, 42
45, 44                                  45, 44
47, 46                                  47, 46
49, 48                                  49, 48
51, 50                                  51, 50
```

Figure 1: from left to right, top to bottom: `prog2-1a.c`, `prog2-1b.c`, compilation/execution and outputs.



```c
#include "stdio.h"
#include "stdlib.h"

int main(int argc, char** argv) {
  double *x = (double *) malloc(sizeof(double));
  double *y = (double *) malloc(sizeof(double));
  int solved = solve(3.0, 1.0, 1.0, 3.0, 2.0, 3.0, x, y);
  if (solved == 1)
    printf("x = %g, y = %g", *x, *y);
  return 0;
}

// Cramer's rule for the given matrix and vector.
int solve(double a, double b, double c, double d, double p, double q, double *x,
 double *y) {
  double D = a * d - b * c;
  if (D == 0.0) return 0;
  double Dx = p * d - b * q;
  double Dy = a * q - p * c;
  *x = Dx / D;
  *y = Dy / D;
  return 1;
}
```

```
fundamental/03 - [master•] » gcc prog2-2.c -o prog2-2
prog2-2.c:7:16: warning: implicit declaration of function 'solve' is invalid in C99
      [-Wimplicit-function-declaration]
  int solved = solve(3.0, 1.0, 1.0, 3.0, 2.0, 3.0, x, y);
               ^
1 warning generated.
fundamental/03 - [master•] » ./prog2-2
x = 0.375, y = 0.875%
fundamental/03 - [master•] »
```

Figure 2: from left to right: `prog2-2.c` and compilation/output.