Universidade de Itaúna - Ciência da Computação
Laboratório de Algoritmos e Estruturas de Dados I
Professor: Thiago Silva Vilela
Solução do Exercício sobre Funções e Structs

Exercício 1

```c
#include <stdio.h>
#include <stdlib.h>

#define TAM 3

typedef struct {
  int dia;
  int ano;
  char mes[30];
} Data;

typedef struct {
  char nome[31];
  int idade;
  char sexo;
  char cpf[12];
  char cargo[31];
  float salario;
  Data nascimento;
  int cod_setor;
} Funcionario;

int main(int argc, char *argv[]) {
  Funcionario funcionarios[TAM];
  int i;

  // Pega as informacoes para preencher o vetor
  for (i = 0; i < TAM; i++) {
    printf("Entrada de dados do funcionario %d\n", i+1);
    printf("\tNome: ");
    scanf("%s", funcionarios[i].nome);
    printf("\tIdade: ");
    scanf("%d", &funcionarios[i].idade);
    printf("\tSexo: ");
    scanf(" %c", &funcionarios[i].sexo);
    printf("\tCPF: ");
    scanf("%s", funcionarios[i].cpf);
    printf("\tCargo: ");
    scanf("%s", funcionarios[i].cargo);
    printf("\tSalario: ");
    scanf("%f", &funcionarios[i].salario);
    printf("\tData de nascimento: ");
    scanf("%d %s %d", &funcionarios[i].nascimento.dia, funcionarios[i].
        nascimento.mes, &funcionarios[i].nascimento.ano);
    printf("\tCodigo do setor: ");
    scanf("%d", &funcionarios[i].cod_setor);
  }

  // Imprime informacoes
  printf("\n\nImprimindo vetor:\n\n");
  for (i = 0; i < TAM; i++) {
    printf("Funcionario %d:\n", i+1);
    printf("\t Nome: %s\n", funcionarios[i].nome);
    printf("\t Idade: %d\n", funcionarios[i].idade);
```

```
54      printf("\t Sexo: %c\n", funcionarios[i].sexo);
55      printf("\t CPF: %s\n", funcionarios[i].cpf);
56      printf("\t Cargo: %s\n", funcionarios[i].cargo);
57      printf("\t Salario: %.2f\n", funcionarios[i].salario);
58      printf("\t Data de nascimento: %d de %s de %d\n", funcionarios[i].
            nascimento.dia, funcionarios[i].nascimento.mes, funcionarios[i].
            nascimento.ano);
59      printf("\t Codigo do setor: %d\n\n", funcionarios[i].cod_setor);
60    }
61
62    return 0;
63  }
```

Exercício 2

```
1   #include <stdio.h>
2   #include <stdlib.h>
3   #include <time.h>
4   #include <string.h>
5
6   #define NUM 3
7
8   typedef struct {
9     int dia;
10    int mes;
11    int ano;
12  } Data;
13
14  typedef struct {
15    int horas;
16    int minutos;
17    int segundos;
18  } Horario;
19
20  typedef struct {
21    char texto[201];
22    Data d;
23    Horario h;
24  } Compromisso;
25
26  int main(int argc, char *argv[]) {
27    Compromisso comps[NUM];
28    int i;
29    srand(time(NULL));
30
31    // Inicializacao dos compromissos
32    for (i = 0; i < NUM; i++) {
33      comps[i].d.dia = rand()%20+1;
34      comps[i].d.mes = rand()%12+1;
35      comps[i].d.ano = rand()%4+2016;
36      comps[i].h.horas = rand()%24;
37      comps[i].h.minutos  = rand()%60;
38      comps[i].h.segundos = rand()%60;
39      strcpy(comps[i].texto, "Compromisso de teste gerado aleatoriamente.")
            ;
40    }
41
42    // Impressao dos compromissos no formato pedido
43    printf("========Lista de Compromissos========\n");
44    for (i = 0; i < NUM; i++) {
45      printf("Compromisso %i:\n", i+1);
```

```
46      printf("\t Data: %d/%d/%d\n", comps[i].d.dia, comps[i].d.mes, comps[i
            ].d.ano);
47      printf("\t Horario: %d:%d:%d\n", comps[i].h.horas, comps[i].h.minutos
            , comps[i].h.segundos);
48      printf("\t Texto: %s\n\n", comps[i].texto);
49    }
50
51    return 0;
52  }
```

Exercício 3

```
1   #include <stdio.h>
2   #include <stdlib.h>
3   #include <math.h>
4
5   typedef struct {
6     double x;
7     double y;
8   } Ponto;
9
10  typedef struct {
11    Ponto superiorEsquerda;
12    Ponto inferiorDireita;
13  } Retangulo;
14
15  double distacia(Ponto a, Ponto b) {
16    double tmp = (b.x - a.x)*(b.x - a.x) + (b.y - a.y)*(b.y - a.y);
17    tmp = sqrt(tmp);
18    return tmp;
19  }
20
21  int estaContido(Ponto a, Retangulo b) {
22    if (b.superiorEsquerda.x < a.x && b.inferiorDireita.x > a.x && b.
          superiorEsquerda.y > a.y && b.inferiorDireita.y < a.y) {
23      return 1;
24    }
25    return 0;
26  }
27
28  Ponto maisProximoOrigem(Ponto a, Ponto b) {
29    Ponto origem = {0.0, 0.0};
30    double distA = distacia(origem, a);
31    double distB = distacia(origem, b);
32    if (distA < distB) {
33      return a;
34    } else {
35      return b;
36    }
37  }
38
39  int main(int argc, char *argv[]) {
40    Ponto a;
41    Ponto b;
42    Ponto maisProximo;
43    Retangulo ret;
44
45    printf("Entre com as coordenadas x e y do primeiro ponto: ");
46    scanf("%lf %lf", &a.x, &a.y);
47
48    printf("Entre com as coordenadas x e y do segundo ponto: ");
```

```
49    scanf("%lf %lf", &b.x, &b.y);
50
51    printf("Entre com as coordenadas x e y do ponto da superior esquerda do
          retangulo: ");
52    scanf("%lf %lf", &ret.superiorEsquerda.x, &ret.superiorEsquerda.y);
53    printf("Entre com as coordenadas x e y do ponto da inferior direita do
          retangulo: ");
54    scanf("%lf %lf", &ret.inferiorDireita.x, &ret.inferiorDireita.y);
55
56    double dist = distacia(a, b);
57    printf("\nDistancia entre os dois pontos: %lf\n\n", dist);
58
59    maisProximo = maisProximoOrigem(a, b);
60    printf("Ponto mais proximo a origem: (%lf, %lf)\n\n", maisProximo.x,
          maisProximo.y);
61
62    if (estaContido(a, ret)) {
63      printf("O ponto (%lf, %lf) esta contido no retangulo!\n", a.x, a.y);
64    } else {
65      printf("O ponto (%lf, %lf) nao esta contido no retangulo!\n", a.x, a.
          y);
66    }
67
68    if (estaContido(b, ret)) {
69      printf("O ponto (%lf, %lf) esta contido no retangulo!\n", b.x, b.y);
70
71    } else {
72      printf("O ponto (%lf, %lf) nao esta contido no retangulo!\n", b.x, b.
          y);
73    }
74    return 0;
75  }
```