UTFPR - Universidade Tecnológica Federal do Paraná
DACOM - Departamento de Computação
BCC31A :: Algoritmos

Exercícios 10 :: Structs

Instruções Gerais

- Faça cada exercício em uma **função distinta**, deixando-as em um único arquivo (main.c).
- Utilize o VSCode ou outro editor para escrever o código. Para compilar pelo terminal:
  $ gcc aquivo.c -o arquivo.bin -std=c99
- Ao final, compacte o projeto (ou somente o arquivo main.c) em ZIP e envie pelo Moodle.
- Funções permitidas: <stdio.h> printf() e scanf(); <string.h> strlen(), strcpy() e strcmp()

1. Escreva uma função que recebe dois pontos x,y (**struct Point**) e devolve a distância entre os mesmos. Utilize sqrt() da lib <math.h> para obter a raiz quadrada. A distância entre (x_1,y_1) e (x_2,y_2) é dada por:

$$d = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}$$

```c
float distance2D(struct Point p1, struct Point p2);

struct Point {
  float x;
  float y;
};
```

Exemplo de uso na main():

```c
int main() {
  struct Point p1 = {-2.0f, 7.5f};
  struct Point p2 = {5.0f, 12.4f};
  float dist = distance(p1, p2);
  return 0;
}
```

2. Considerando a estrutura Point do exercício 1, escreva uma função que recebe uma lista (vetor) de pontos, bem como seu tamanho, e informa a distância a cada dois pontos consecutivos.

```c
float distanceVector(int n, struct Point v[n]);
```

3. Escreva uma função que recebe como parâmetro uma data de nascimento (**struct Date).** A função deve calcular e retornar uma idade com base na data atual. Para obter a data atual, utilize a função **get_current_date()** (ver anexo).

```c
struct Date {
  int day; // 1-31
  int mon; // 1-12
  int year; // 4 dígitos (YYYY)
};
```

4. Escreva uma função que recebe um pessoa (Person) e imprime seu nome, a quantidade de palavras no nome e sua idade (com base na data de nascimento e na data atual). Para obter a data atual, utilize a função **get_current_time()**, utilizada no exercício anterior.

```c
void printData(struct Person p);

struct Person {
  char name[50];
  int day; // dia nascimento
  int mon; // mês nascimento
  int year; // ano nascimento
};

Exemplo de uso na main():
int main() {
  struct Person p = {"Jovenslau da Silva Sauro", 5, 12, 1999};
  printData(p);
  return 0;
}
```

5. Considerando a struct Person do exercício anterior, escreva uma função que recebe um vetor de pessoas e seu tamanho. A função deve imprimir nome, data de nascimento e idade de cada pessoa.

```c
void printPersonVector(int n, Person v[n]);
```

6. Escreva uma função que recebe uma lista de empregados (struct Employee) e seu tamanho. A função deve calcular e imprimir: o total dos salários, a média dos salários e as médias dos salários por tipo de empregado (“Developer”, “Designer”, “Manager” ou “Support”).

```c
void printReport(int n, struct Employee v[n]);

struct Employee {
  char name[50];
  float salary;
  char type[50];
  /*  D - Developer
      E - Designer
      M - Manager
      S - Support  */
};
```

7. Escreva uma função que imprime os dados (nome, email e telefones) de um contato (Contact).

```c
void printContact(struct Contact c);

struct Phone {
  char ddd[3];
  char number[10];
  char type;
 //H - Home / W - Work / M - Mobile
};
```

```c
// Exemplo de struct aninhada
struct Contact {
  char name[50];
  char email[70];
  struct Phone phone1;
  struct Phone phone2;
};
```

Exemplo de uso na função main():

```c
int main() {
  Contact c =
    {"Joao", "joao@gmail.com",
      {"44", "30456782", "work"},
      {"44", "998769034", "mobile"}
    };

  // Exemplo de acesso ao campo telefone 1 do contato
  printf("phone 1: (%3s) %s\n", c.phone1.ddd, c.phone1.number);

  printContact(c);
  return 0;
}
```

8. Considerando as structs Contact e Phone do exercício anterior, escreva uma função que recebe um vetor de contatos, seu tamanho e um tipo de telefone (‘H‘, ‘W‘, ou ‘M‘). A função deve imprimir os nomes dos contatos que possuem algum telefone do tipo informado.

```c
void printContactByPhoneType(int n, struct Contact v[n], char phone_type);
```

9. Defina as estruturas abaixo, relativas a uma Agenda de Eventos:
   a. Data: composto por dia, mes e ano;
   b. Evento: composto por data, duração em minutos e descrição.

10. De acordo com as estruturas definidas no exercício 8, escreva uma função que recebe um Evento e imprime seus dados.

11. De acordo com as estruturas definidas no exercício 8, escreva uma função que recebe um vetor de Eventos, bem como seu tamanho, e o inicializa com dados aleatórios. A descrição do Evento deve ser sorteado entre um dos seguintes: “casa”, “trabalho”, “estudos” ou “lazer”.

12. De acordo com as estruturas definidas no exercício 8, escreva uma função que recebe um vetor de Eventos, bem como seu tamanho, e imprime os dados de todos os eventos.

13. De acordo com as estruturas definidas no exercício 8, escreva uma função que recebe um vetor de Eventos, bem como seu tamanho, e imprime os dados de todos os eventos que possuam a duração mínima e o texto informado.

Exemplo de uso:

```c
// a função imprimirá todos os eventos com descrição "trabalho" e com duração de,
// ao menos, 30 minutos
printEventsFiltered(50, list_of_events, 30, "trabalho");
```

ANEXO: função **get_current_time()** para obtenção da data atual + programa de teste.

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

struct Date {
  int day; // 1-31
  int mon; // 1-12
  int year; // 4 dígitos (YYYY)
};

struct Date get_current_date() {
    time_t now;
    //retorna a contagem de segundos desde 01/01/1970 (padrão Unix)
    time(&now);

    // converte time para calendário e retorna uma struct contendo
    // os campos dia, mês, ano, hora, minuto e segundo
    struct tm* p_time = localtime(&now);

    // copia os dados do struct tm em Heap para um struct Date e o devolve
    struct Date d;
    d.day = p_time->tm_mday;
    d.mon = p_time->tm_mon + 1;      // na struct, mês 0-11
    d.year= p_time->tm_year + 1900;  // na struct, conta ano a partir de 1900

    return d;
}

// Imprime hora e data atuais
int main(void) {
    struct Date d = get_current_date();

    printf("Data: %02d/%02d/%d\n", d.day, d.mon, d.year);

    return 0;
}
```