Atividade Prática 01

Estruturas de Dados I – 2022/2

Prof. Dr. Thiago L. T. da Silveira
tltsilveira@inf.ufrgs.br

Enunciado Geral

Crie um Tipo Abstrato de Dados (TAD) chamado `PontoEletronico` em linguagem C. Este TAD deve simular o funcionamento de um ponto eletrônico para registro de horário de entrada e saída de funcionários de uma empresa/indústria, e deve contabilizar a carga horária diária. Você deve criar uma estrutura (`struct`) `PontoEletronico` com variáveis de controle (campos) que permitam resolver o problema. Você tem liberdade para definir quais campos usar. No entanto, o TAD `PontoEletronico` deve possuir as seguintes operações (funções):

`iniciaPontoEletronico`, que recebe um ponteiro para uma estrutura `PontoEletronico` e inicializa suas variáveis de controle. Os valores iniciais dos campos dessa estrutura devem atender às demandas das demais funções.

`registraPonto`, que recebe um ponteiro para uma estrutura `PontoEletronico` e uma hora e um minuto (ambos inteiros) correspondendo ao horário a ser registrado. Esta função deve receber apenas esses argumentos. A função `registraPonto` deve guardar a hora e minuto de entrada de um turno de trabalho caso o registro de ponto seja referente à chegada do funcionário no trabalho. A função `registraPonto` deve contabilizar o tempo de trabalho caso o registro de ponto seja referente à saída do funcionário do trabalho. Considere que funcionários sempre registram pontos de chegada antes de pontos de saída. Assuma horários em formato de 24h (de 00:00h à 23:59h) e que não há turnos envolvendo dois ou mais dias. Assuma também que o funcionário sempre vai registrar horários corretos (valores correspondendo a minutos e horas em intervalos adequados).

`imprimeTotalTrabalhadoDia`, que recebe uma estrutura `PontoEletronico` e imprime na tela o total de horas e minutos trabalhados no dia. O formato de impressão deve ser "horas:minutos".

Defina a estrutura `PontoEletronico` e as operações (funções) necessárias (descritas acima). Organize o TAD em arquivos de interface (`pontoEletronico.h`) e de implementação (`pontoEletronico.c`). Copie e cole o código abaixo em um arquivo de nome `programa.c` para testar sua implementação do TAD `PontoEletronico`. As funções `iniciaPontoEletronico`, `registraPonto` e `imprimeTotalTrabalhadoDia` devem ser condizentes com as chamadas do programa de exemplo (abaixo). Nenhuma função tem retorno.

Dicas:

- Compreenda (idealmente com auxílio de papel e caneta) o "algoritmo" necessário para computar a diferença entre dois horários dados no formato de 24h. Comece a programar apenas depois disso.
- Identifique a importância de cada campo da estrutura `PontoEletronico` e cada função (e o motivo dela receber ou não um ponteiro para estrutura). Por exemplo, por que temos a função `iniciaPontoEletronico`? O que há em um campo de uma estrutura não "inicializado"?
- Lembre-se que, para acessar um campo de uma estrutura, usamos o operador `.` (ponto). Se quisermos acessar um campo de um ponteiro para estrutura, então devemos usar o operador `->` (sinal de menos seguido do sinal de maior). Retome os *slides* 7 a 9 da Aula 04 para exemplos.

Programa Exemplo:

```
#include <stdio.h>
#include "pontoEletronico.h"

int main(){

      PontoEletronico pEletronico;
      iniciaPontoEletronico(&pEletronico);

      registraPonto(&pEletronico, 8, 10); // Chegada ao trabalho
      registraPonto(&pEletronico, 12, 0); // Intervalo para almoco

      registraPonto(&pEletronico, 13, 10); // Volta do almoco
      registraPonto(&pEletronico, 17, 20); // Saida do trabalho

      imprimeTotalTrabalhadoDia(pEletronico); // Deve imprimir 8:0

      return 0;
}
```

Bônus

Implemente o comportamento do TAD `PontoEletronico` de tal forma que seja possível registrar mais de dois turnos de trabalho. Você, ainda assim, pode considerar que um turno não envolve dois dias (isto é, a entrada e a saída do turno de trabalho acontecem dentro do período de 24h).

Instruções

- Essa Atividade deve ser resolvida de forma individual e em sala de aula.
- Essa Atividade deve ser entregue via Moodle, em local indicado, até a data limite pré-estipulada.
- Lembre-se submeter seu(s) código(s) em arquivo compactado (`.zip` ou `.rar`) com nome `APX-<nome_e_ultimo_sobrenome>.zip`. Troque `X` pelo identificador desta Atividade Prática.
- Seu programa deve ser compilável. Priorize a apresentação de versões de seu código que não apresentam erro de execução ainda que incompletas. Documentação e legibilidade do código fazem parte da avaliação.