Trabalho Prático 01 – AEDS 1 – Trios

Professora: Thais R. M. Braga Silva

Valor: 10 pontos

Data de Entrega: 16/03/2021 (Entrevistas: 16/03 e 18/03)

Forma de Entrega: PVANet (formato .zip, .rar ou .tar.gz)

Tendo em vista o aumento esperado dos encargos didáticos no próximo ano, o Campus UFV-Florestal resolveu desenvolver um sistema automatizado de gerenciamento de tempo para os professores. A especificação do sistema segue abaixo.

Em linhas gerais, esse é um sistema que permite a criação de uma agenda para cada professor. Cada agenda é composta por um conjunto de compromissos. Após criada, a agenda poderá ser visualizada ou alterada pelo professor, isto é, os compromissos da agenda podem ser mostrados ou seus detalhes modificados.

Tipos Abstrato de Dados

Você deve implementar, obrigatoriamente, os TADs definidos a seguir:

1. TAD Compromisso

A agenda de um professor possui compromissos, definidos a partir de um identificador numérico, bem como informações adicionais associadas: data (ano, mês e dia), hora (horas e minutos), duração (em minutos) e descrição. Você pode assumir que a descrição é uma cadeia de caracteres, com um tamanho máximo M = 100. Também assuma que a duração de um compromisso nunca deverá fazer com que o mesmo termine em um dia diferente da data inicialmente atribuída ao mesmo, ou seja, um compromisso sempre começa e termina no mesmo dia.

Cada compromisso tem também um grau de prioridade que deve ser um número inteiro entre 1 (maior prioridade) e 5 (menor prioridade). Um professor deve poder alterar (aumentar ou diminuir) o grau de prioridade de qualquer compromisso.

Um compromisso deve suportar, no mínimo, as seguintes operações:

• inicializaCompromisso(prioridade, data, hora, duração e descrição): inicializa um compromisso com os valores informados. Esta função deve atribuir um identificador numérico único para o compromisso e retorná-lo como resultado.

• alteraPrioridade(novaPri): altera a prioridade de um compromisso com um novo valor passado como parâmetro.

• retornaPrioridade(): retorna a prioridade de um compromisso.

• temConflito(compromisso1, compromisso2): esta operação deve retornar se há ou não conflito nos horários dos dois compromissos passados como parâmetros. Um conflito ocorre caso o horário de realização de um compromisso, incluindo sua duração, possua interseção com o horário de ocorrência do outro.

• imprimeCompromisso(): imprime todas as informações associadas a um compromisso seguindo a ordem identificador, prioridade, data, hora, duração e descrição.

2. TAD Agenda

O sistema sendo desenvolvido deve permitir a criação de agendas, uma para cada professor.

Assim, cada agenda deve conter

• um identificador único do professor,

• o nome do professor,

• o ano,

• uma sequência de compromissos.

Um professor deve ser capaz de, no mínimo, realizar as seguintes operações com sua agenda:

• criaAgenda(idProf, nome, ano): cria uma nova agenda para o professor com identificador e nome informados. O ano também é passado como parâmetro. Esta operação deve criar uma agenda vazia, sem nenhum compromisso.

• recuperaAgenda(data): recebe uma data, e retorna uma mensagem na tela com o nome do professor, o ano e o número de compromissos na agenda com data após a data fornecida.

• insereCompromisso(prioridade, data, hora, duração, descrição): recebe as informações associadas de um novo compromisso, que deve ser inicializado (chamando a operação respectiva no TAD Compromisso) e inserido na agenda. A operação deve imprimir uma mensagem informando sucesso ou falha. Em caso de sucesso, o identificador do compromisso deve ser impresso. O sistema não deve permitir a inserção de um novo compromisso que esteja em conflito com outro já cadastrado. Nesse caso, ele deverá retornar uma mensagem indicando falha na operação. Lembre-se que o TAD compromisso possui um método que verifica conflito entre dois compromissos.

• removeCompromisso(idCompromisso): recebe o identificador de um compromisso. Esta operação deve remover o compromisso identificado da agenda do professor, caso ele exista.

• imprimeAgenda(): esta operação deve imprimir todos os compromissos agendados do professor, ordenados pela prioridade (em ordem decrescente de prioridade).

• retornaNCompromissos(): retorna o número total de compromissos na agenda.

3. Outros TADs

Além dos dois TADs mencionados acima, cada grupo deverá definir dois outros TADs e implementá-los.

Sistema de Gerenciamento de Tempo

O sistema de gerenciamento de tempo a ser desenvolvido deve permitir a criação, atualização e visualização das agendas de todos os P professores do campus.

O sistema deverá ter duas formas de entrada: interativa e por arquivo. A forma interativa servirá para testes pequenos, simulando a situação em que o professor estivesse efetivamente utilizando o sistema. A forma por arquivo servirá essencialmente para realização de testes maiores e mais rápidos.

Na forma interativa o sistema deve permitir que o usuário (isto é, um professor) defina qual operação deseja realizar em cada interação. Em outras palavras, o sistema deve fornecer um menu para que o usuário interaja com o sistema e eventualmente saia do mesmo.

Escolhida a operação, o sistema deverá solicitar que o usuário entre com as informações necessárias para a sua realização e imprima os resultados (se houver) na tela.

O menu não precisa ser sofisticado, bastando uma interface ASCII simples com código de escolha de cada operação que o usuário pode fazer em cada instante.

O sistema deve permitir que um professor cuja agenda já foi cadastrada faça login, fornecendo seu identificador de professor. Nesse caso, o menu de opções não mostra mais o item para criação de agenda, apenas para atualização e visualização.

Na forma de entrada por arquivo, o sistema deverá ler de um arquivo de entrada todas as informações necessárias para criar as agendas e seus respectivos compromissos. Em seguida, disponibilizará o menu de login e as operações de consulta ao sistema de gerenciamento de tempo, da mesma forma como foi feito para a forma de entrada interativa.

O formato do arquivo será fornecido pela professora juntamente com a especificação do TP. Um arquivo exemplo será utilizado como forma de demonstrar o formato, bem como poderá ser utilizado pelos alunos para testes.

Ao final, você deverá submeter todos os arquivos de código compactados através do PVANet. Coloque todas as suas considerações em forma de comentários dentro do código.

Em particular, atente para:

- **ATENÇÃO: TODAS AS LISTAS DESSE TP DEVEM SER IMPLEMENTADAS COM USO DE LISTA ENCADEADA COMO ESTRUTURA DE DADOS**
- Cada TAD deve, necessariamente, ser implementado em arquivos separados, com uso de arquivo cabeçalho (.h) e arquivo de implementação (.c)

- A implementação de lista encadeada deve ser tal qual vista em sala de aula
- O programa deve estar bem indentado e comentado
- **Trabalhos copiados serão penalizados. Trabalhos em atraso não serão aceitos.**

ATENÇÃO: Soluções que não correspondam à implementação de um Tipo Abstrato de Dados serão duramente penalizadas por não atenderem a especificação.

O que deve ser entregue:

- Todo código fonte produzido, incluindo os arquivos de cabeçalho
- Uma pequena porém completa documentação, descrevendo o objetivo do trabalho, o projeto do sistema implementado, as principais decisões de projeto, os módulos desenvolvidos, os métodos implementados e a conclusão. O formato para a entrega da documentação é necessariamente PDF
- Fazer um zip ou tar.gz de todos os arquivos, nomeá-lo com o nome e número de matrícula de cada integrante do grupo, bem como número do TP, e submetê-lo apenas uma vez pelo PVANet.

Como será a avaliação:

- Entrevista com o monitor: o monitor avaliará a compilação e execução do trabalho, bem como conduzirá alguns testes sobre o código entregue. Perguntas serão feitas sobre as decisões de projeto e detalhes de implementação;
- Avaliação da professora: a professora avaliará o código e a documentação entregue e, juntamente com a avaliação fornecida pelo monitor, deliberará sobre a nota final de cada grupo.