Trabalho 1
Programação Orientada a Objetos

Leia atentamente TODO o enunciado do trabalho (a especificação do problema e os detalhes sobre a confecção, submissão e avaliação do trabalho). Uma leitura inapropriada do enunciado pode ser extremamente danosa a sua nota.

O professor Felisberto é um homem muito ocupado. Tem, em sua agenda, uma série de compromissos que ele deve atender. De tantos compromissos que tem, resolveu contratar programadores para desenvolver um sistema de filtro de compromissos. Seu trabalho é desenvolver este sistema.

1 Descrição do problema

1.1 Os compromissos

A agenda do professor possui diversos tipos de compromisso:

- Aulas para turmas de graduação e pós-graduação: inclu data, hora, duração (em minutos), disciplina ministrada e se é de graduação, especialização ou mestrado;
- Orientação de alunos de graduação e pós-graduação: inclui data, hora, duração (em minutos), nome do orientado e se é de graduação, especialização ou mestrado;
- Reuniões de departamento: inclui data, hora e duração (em minutos);
- Eventos: inclui data, hora, duração (em dias), nome do evento e local;
- Compromissos particulares: deseja-se saber data, hora, duração (em minutos) e o motivo.

Para efeito de relatório, cada compromisso possui um identificador e uma descrição. O identificador é numérico e será lido do arquivo de entrada. A descrição segue um padrão que varia de acordo com o tipo, segundo a tabela abaixo. Substitua os conteúdos entre <> por dados do compromisso.
Para cada compromisso, Felisberto atribui um grau de prioridade. Além disso, cada tipo de compromisso possui um "fator multiplicador" de prioridade, que representa uma classificação de importância dos tipos de compromisso, ou seja, os que tem menor grau de prioridade são

Tipo de Evento	**Descrição**
Aulas	Aula de <nome da disciplina>
Orientação	Orientação de <nome do aluno>
Reunião de Departamento	Reunião de Departamento
Eventos	<nome do evento>
Compromissos Particulares	<motivo>

aqueles que o professor está mais apto a cancelar ou, se possível, adiar. Os fatores de multiplicação são dados na tabela abaixo:

Tipo de Evento	**Fator Multiplicador**
Aulas	2x
Orientação	1x
Reunião de Departamento	4x
Eventos	3x
Compromissos Particulares	2x

Para todos os compromissos da agenda, além de ddeterminar a prioridade, deve-se também indicar se o compromisso é adiável ou não, sendo que aulas e eventos nunca são adiáveis. Compromissos inadiáveis (exceto aulas e eventos, que já o são por natureza), devem receber automaticamente +1x em seu fator multiplicador. Ou seja, um compromisso particular adiável tem um fator multiplicador 2x, já um inadiável tem fator (2+1)x = 3x.

1.2 O filtro

O professor Felisberto gostaria de ter um filtro que, dados todos os eventos que deseja participar, com seus momentos de ocorrência e prioridades, localize compromissos conflitantes e indique, em relatórios, quais deles devem ser cancelados ou adiados.

Seu programa deve separar em quatro relatórios diferentes, **a)** um para compromissos confirmados, **b)** outro para os que devem ser adiados, **c)** outro para os que devem ser cancelados e **d)** um último com os dados completos de todos os compromissos.

a) No relatório de compromissos confirmados devem constar todos os compromissos que não conflitaram com nenhum outro ou que, se conflitaram, tiveram a maior prioridade e passaram pelo filtro. De cada compromisso deve-se apresentar data, hora, identificador e descrição. Os compromissos devem estar ordenados por data e hora em ordem crescente;

b) No relatório de compromissos adiados encontram-se os compromissos adiáveis que conflitaram com algum outro compromisso e que não tiveram a maior prioridade, ou seja, terão que ser adiados, pois outro compromisso mais importante deve ser atendido. De cada compromisso deve-se apresentar identificador, descrição, prioridade e identificador

do compromisso mais importante que forçou o adiamento deste. Os compromissos devem estar em ordem decrescente de prioridade;

c) No relatório de compromissos cancelados encontra-se os compromissos inadiáveis que conflitaram com algum outro compromisso e que não tiveram a maior prioridade, ou seja, terão que ser cancelados, pois outro compromisso mais importante deve ser atendido. De cada compromisso deve-se apresentar identificador, descrição, prioridade e identificador do compromisso mais importante que forçou o cancelamento deste. Os compromissos devem estar em ordem crescente de duração;

d) No relatório completo, devem ser exibidos todos os compromissos e todos os dados referentes aos mesmos. O relatório deve encontrar-se ordenado por identificador, em ordem crescente.

Além desses quatro relatórios, seu programa deve gerar um outro arquivo, chamado resultado.txt, contendo como único valor a soma das durações (em minutos) do i-ésimo elemento do relatório do item a), do j-ésimo elemento do item b), do k-ésimo elemento do item c) e do m-ésimo elemento do item d). Os valores i, j, k e m são fornecidos em um arquivo de entrada chamado posicoes.txt.

Nos itens b) e c), caso haja empate, isto é, dois ou mais compromissos tenham a mesma prioridade (obtida multiplicando o grau de prioridade pelo fator multiplicador) ou a mesma duração, utilize os seguintes critérios de "desempate":

- É mais importante o compromisso que iniciar primeiro;
- Caso ainda haja empate, o tipo de compromisso deve desempatar na seguinte ordem de prioridade (da maior para a menor): reunião, evento, aula, particular, orientação;
- Caso ainda haja empate, é mais importante o que for primeiro em ordem alfabética de descrição.

Tais critérios solucionam todos os empates, visto que não pode haver dois compromissos com a mesma descrição iniciando no mesmo horário.

Nas seções seguintes são dados os padrões de entrda e saídad de dados e exemplos.

2 Formatos de Entrada e Saída

2.1 Entrada de dados

O arquivo principal de entrada de dados será chamado de *agenda.txt* e conterá todos os compromissos, um seguido do outro, sem qualquer ordem específica. Todos os compromissos começam com o tipo e identificador e os dados, na sequência, dependem do tipo.

Cada compromisso deve ser separado do compromisso seguinte por uma linha em branco.

Deve haver duas linhas em branco ao final do arquivo, logo após o último compromisso descrito.

A tabela a seguir especifica o padrão de entrada de dados para cada compromisso.

Tipo de Evento	**Padrão de Entrada de Dados**
Aulas	A<identificador> <data> <hora> <duração> <nome da disciplina> <Gradução | Especialização | Mestrado> <grau de prioridade>
Orientação	O<identificador> <data> <hora> <duração> <adiável> <nome do orientado> <Gradução | Especialização | Mestrado> <grau de prioridade>
Reunião de Departamento	R<identificador> <data> <hora> <duração> <adiável> <grau de prioridade>
Eventos	E<identificador> <data> <hora> <duração> <nome do evento> <local> <grau de prioridade>
Compromissos Particulares	P<identificador> <data> <hora> <duração> <adiável> <motivo> <grau de prioridade>

Um exemplo pode ser encontrado no arquivo anexo exemplo/agenda.txt. É importante salientar que não devem haver espaços em branco ou linhas em branco a mais do que a especificação, do contrário a correção pode não funcionar.

O arquivo posicoes.txt conterá valores inteiros positivos (as posições começam a ser contadas a partir da posição 1) de i, j, k e m. Os valores de i, j, k e m se encontrarão posicionados nesta sequência, cada um em uma linha. Um exemplo pode ser encontrado no arquivo anexo

exemplo/posicoes.txt.

2.2 Saída de Dados

Os relatórios gerados devem ser escritos em arquivos com os seguintes nomes:

Relatório	**Nome do Arquivo**
Compromissos confirmados	relatconfirmados.txt
Compromissos adiados	relatadiados.txt
Compromissos cancelados	relatcancelados.txt
Relatório completo	relatcompromissos.txt

O padrão na construção do relatório depende do tipo do relatório, como nos mostra a tabela a seguir.

Relatório	**Nome do Arquivo**
Compromissos confirmados	<data> <hora> <identificador> <descrição>
Compromissos adiados	<identificador> <descrição> <prioridade> <identificador do compromisso que adiou este>
Compromissos cancelados	<identificador> <descrição> <prioridade> <identificador do compromisso que adiou este>
Relatório completo	<identificador>: <descrição> Início: <data> <hora> Fim: <data fim> <hora fim> Prioridade: <prioridade>

O arquivo resultado.txt contém apenas um valor inteiro positivo.

É ainda mais importante que os relatórios sejam gerados rigorosamente dentro do padrão. Assim como no padrão de entrada, os compromissos devem estar separados por uma linha em branco e não deve haver linhas em branco no final dos arquivos, pois serão usados na correção automática do trabalho.

Exemplosp odem ser encontrados em anexo nos arquivos de exemplo relativos a cada relatório.

2.3 Formatos

Os formatos devem seguir os padrões abaixo:

Dado	**padrão**
<identificador>	Número inteiro com 6 algarismos. Preceder com zeros se for necessário. Ex.: 00153.
<data>	Formato dd/mm/aaaa. Ex.: 22/09/2025.
<hora>	Formato hh::mm. Ex.: 03:45.
<duração>	número inteiro.
<nome da disciplina>	String de tamanho máximo 50.
<Graduação | Especialização | Mestrado >	Exatamente uma dessas três opções.
<grau de prioridade>	Número inteiro.
<adiável>	true ou false
<nome do orientado>	String de tamanho máximo 50.
<nome do evento>	String de tamanho máximo 50.
<local>	String de tamanho máximo 50.
<motivo>	String de tamanho máximo 50.
<data fim>	Formato dd/mm/aaaa. Ex.: 22/09/2025.
<hora fim>	Formato hh::mm. Ex.: 03:45.

3 Requisitos de implementação

- Modularize seu código adequadamente. O uso de variáveis globais é proibido, mas constantes globais são permitidas;
- Crie códigos claros e organizados. Utilize um estilo de programação consistente. Comenten seu código extensivamente;
- Seu programa deverá ler os dados de entrada de um arquivo texto chamado agenda.txt e de outro chamado posicoes.txt que se encontram na mesma pasta que o arquivo executável;
- O programa deve utilizar um único subprograma de ordenação genérico que será usado para todos os tipos de ordenações requeridas nos relatórios;
- Você deve implementar uma lista encadeada genérica para usar no armazenamento dos registros de cada compromisso da agenda nos trabalhos de C (lista de void*) e C++ (lista de template). Isso signifca que você não deve utilizar bibliotecas de listas para estas liguagens;
- Você deve criar uma hierarquia de registros (estruturas ou classes) para representar os diferentes tipos de compromissos da agenda.

4 Condições de entrega

TBD - github classroom.

5 Avaliação

- Utilize o valgrind para verificação de vazamentos de memória;
- Seu trabalho deve conter um Makefile;
- Seu trabalho deve conter um README.md identificando o autor do trabalho: nome e número de matrícula;
- TBD.