	PUC-MG: Pontifícia Universidade Católica de Minas Gerais
	Curso: Engenharia de Software
	Disciplina: Programação Modular
	Professor(a): Pedro Pongelupe Lopes
	Semestre: 2024.1
	Aluno: | **Matrícula:**

Exercício

Questão 1 Uma empresa de TI envolvida com políticas públicas prestará serviço a organizações diversas. O software desta empresa possibilitará que uma organização interessada crie ações cívicas para impulsionar políticas de interesses de um grupo de cidadãos. Uma ação cívica pode ser uma petição ou uma campanha.

No caso de uma petição, as pessoas interessadas assinarão a ação. Este conjunto de signatários servirá, posteriormente, para gerar um texto impresso com o texto da petição e os dados de cada pessoa que a apoiou.

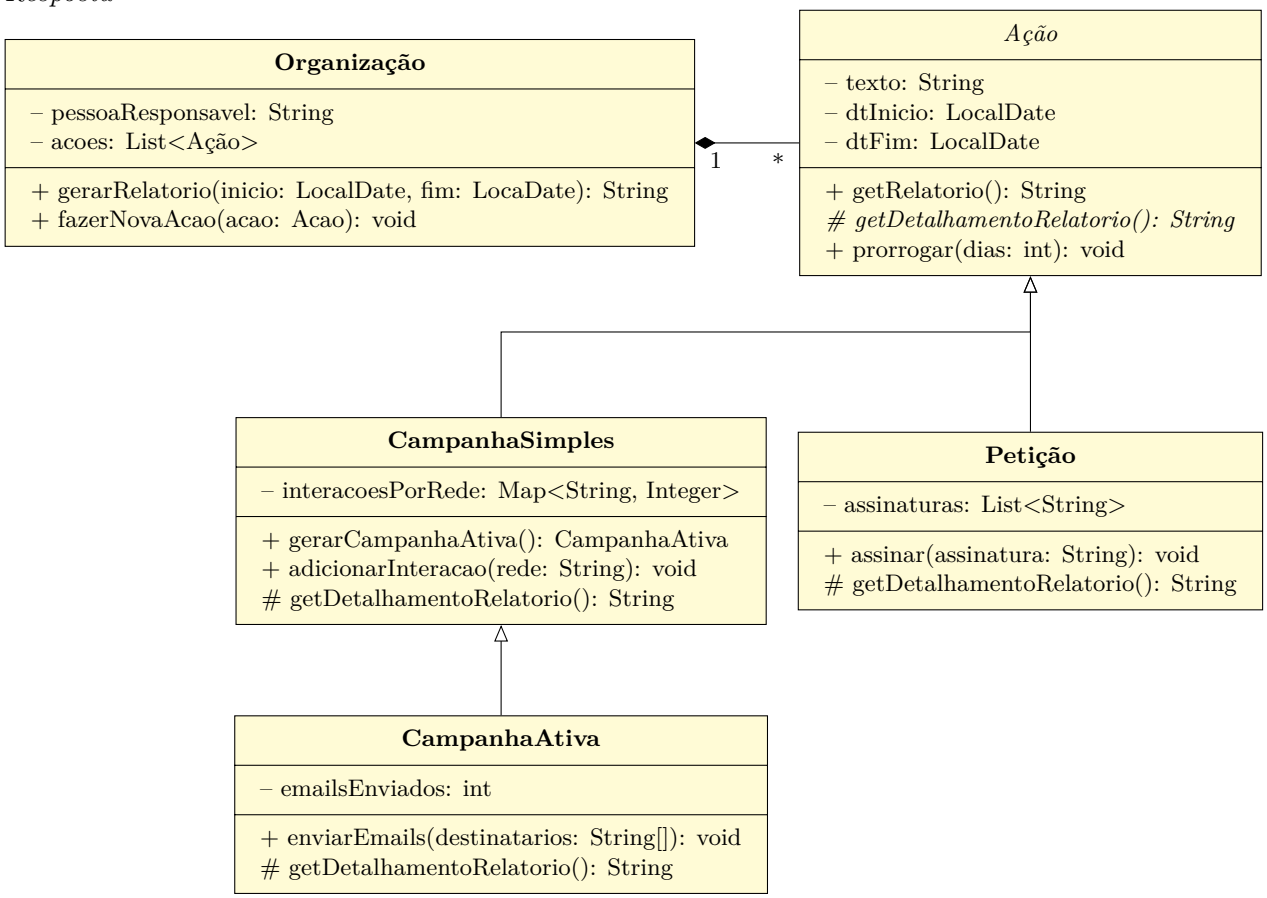
Já as campanhas são divulgadas em redes diversas: Lifegram, Peoplebook e Twinger. Nestas redes, cada interação com a campanha é registrada. As campanhas simples apenas são publicadas e, quando encerradas, geram um relatório contendo o número de interações em cada rede. Já as campanhas ativas são publicadas em uma rede e, além disso, possibilitam que o mesmo texto seja enviado por email para um ou mais destinatários. O relatório desta campanha contabiliza as interações e a quantidade de emails enviados. Dependendo do sucesso da campanha, a organização promotora pode resolver transformar uma campanha simples em uma campanha ativa.

Qualquer tipo de ação cívica precisa ter datas de início e fim, podendo ser prorrogada por alguns dias se assim for o interesse organização promotora. Portanto, estes dados devem estar em qualquer relatório gerado para a organização. Uma organização precisa ter uma pessoa responsável pela criação das campanhas e para ser o contato que receberá os relatórios das ações promovidas pela organização.

Como líder da equipe, que ainda é pequena e está em formação, suas tarefas são:

(a) Utilizando todos os conceitos vistos até hoje na disciplina, **modele um diagrama de classes UML** para o problema proposto. O modelo deve incluir **classes, relacionamentos, atributos e métodos** necessários para resolver completamente o problema. **Não é necessário incluir** construtores ou métodos get/set, mas indique as visibilidades de métodos e atributos.

Resposta



(b) Considerando seu modelo em (a), escreva o código dos métodos para:

- Transformar uma campanha simples em ativa.

Resposta

```
public CampanhaAtiva gerarCampanhaAtiva() {
  return new CampanhaAtiva(this);
}
```

- Gerar relatórios de todas as ações promovidas por uma organização entre duas datas.

Resposta

```
public String gerarRelatorio(LocalDate inicio, LocalDate fim) {
  return acoes
    .stream()
    .filter(a -> (a.getDtInicio().equals(inicio) || a.getDtInicio().after(inicio))
                 && (a.getDtFim().equals(fim) || a.getDtFim().before(fim)))
    .map(Acao::getRelatorio)
    .reduce("Pessoa responsavel: ", (acc, el) -> acc.concat(el));
}

// Acao
public String getRelatorio() {
  return """
         ACAO
      ------------------------
      Data Inicio: %s
      Data Fim: %s
      Texto: %s
      Detalhes: %s
    """.formatted(dtInicio, dtFim, texto, getDetalhamentoRelatorio());
}

// Peticao
@Override
public String getDetalhamentoRelatorio() {
  return assinaturas
         .stream()
         .reduce("", (acc, el) -> acc.concat("\n" + el));
}

// CampanhaSimples
@Override
public String getDetalhamentoRelatorio() {
  return interacoes
         .entrySet()
         .stream()
         .reduce("", (acc, el) ->
                  acc.concat("rede social %s \n interacoes: %d \n"
                             .formatted(el.getKey(), el.getValue()));
}

// CampanhaAtiva
@Override
```

```
public String getDetalhamentoRelatorio() {
  return super.getRelatorio()
              .concat("emails enviados: " + emailsEnviados);
}
```