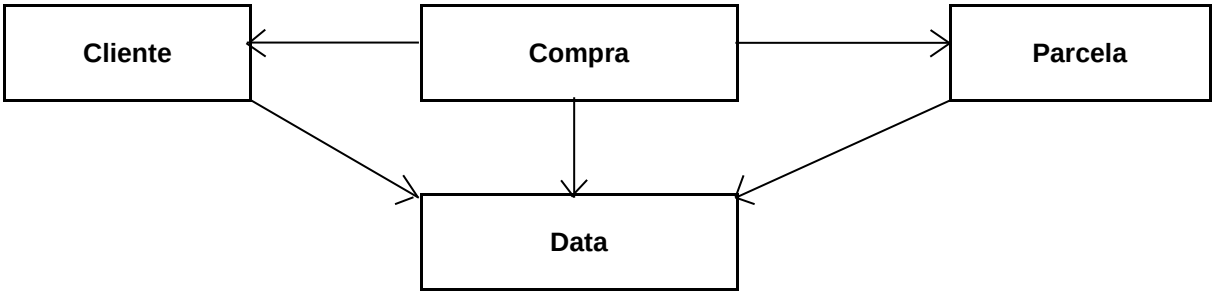
Em grupos de até 2 alunos. O trabalho é extraclasse – **NÃO DEVE SER FEITO DURANTE AS AULAS** salvo com autorização do professor.

Simulação (simplificada) de um *sistema de controle de vendas.* O relacionamento entre as principais classes aparece abaixo:



Programe as três classes seguintes e a classe de teste. Use o seguinte comentário no início do texto fonte de cada classe:
/ Alunos : xxxxxxxxxxxxx e xxxxxxxxxxx Trabalho GA Lab 1 turma XX 2016/2 */**

Classe **Data**

É dada. Está no arquivo Data.java que pode ser baixado diretamente da comunidade.

Classe **Parcela**

- Cada objeto do tipo **Parcela** representa uma prestação resultante de uma compra parcelada.

Atributos privados (somente estes):
- cliente – objeto do tipo **Cliente**, que indica quem fez a compra
- data de vencimento – objeto do tipo **Data**
- valor original – é o valor da parcela, sem qualquer acréscimo
- valor final – é o valor original acrescido ou não de juros por atraso no pagamento
- situação – tipo **char**. Os valores possíveis são:
  N – parcela ainda não venceu, nem foi paga;
  Q – parcela já foi quitada (paga);
  A - parcela ainda não foi paga e está em atraso.

Construtor: um só, com três parâmetros: tipo **Cliente**, tipo **Data** (para informar a data de vencimento) e um terceiro com o valor original da parcela. Inicializar a situação como N.

Métodos:
+ *registraAtraso* – instancia a data de hoje e altera a situação para A, se a data de vencimento já passou em relação à data de hoje. Obviamente, esta alteração só será feita se a parcela estava na situação N. Método retorna true ou false, para indicar se houve ou não a mudança da situação.

+ *paga* – método que será chamado por ocasião do pagamento da parcela. Recebe como parâmetro a data em que está sendo feito o pagamento. Se estiver sendo paga com até 5 dias de atraso, pagará juros de 1% sobre o valor original; de 6 a 15 dias de atraso, juros de 1,5%; atraso acima de 15 dias, juros de 2,5%. O método deve atualizar os atributos situação e valor final, desta classe, além de registrar o pagamento da mesma no objeto **Cliente,** chamando o método daquela classe adequado para isso. Ao final, o método deve retornar o valor dos juros.

+ traduz a situação – este método simplesmente retorna o valor da situação na forma de um String: *não venceu ainda, em atraso, quitada.*

+ *exibeDados* – Recebe o número da parcela como parâmetro. Mostra na tela do usuário os dados atuais da parcela, começando com o número dela, tudo numa única linha, contendo o nome do cliente, a data de vencimento da parcela, o valor original e a situação (por extenso).

+ métodos *get* e *set* que forem necessários a critério do grupo.

Classe **Cliente**

Atributos privados (só estes):
- nome
- data de nascimento – objeto do tipo **Data**
- valor da penúltima compra
- valor da última compra
- saldo devedor – corresponde à soma de todas as parcelas do cliente que ainda não foram pagas.

Construtor: recebe dois parâmetros: o nome e a data de nascimento (este é do tipo **Data**).

Métodos:
+ *fazCompra* – chamado sempre que o cliente fizer uma compra à vista. Tem um parâmetro com o valor final da compra e atualiza devidamente os atributos penúltima e última compra.

+ *fazCompra* – sobrecarga do método anterior, que só será chamado quando a compra for parcelada. Tem dois parâmetros, o valor final da compra e a soma das parcelas que deverão ser pagas no futuro. Deve chamar o método anterior para atualizar os dois atributos de penúltima e última compra, e atualizar o atributo saldo devedor. Cuidado para não perder o saldo devedor já existente.

+ paga uma parcela – chamado sempre que é feito o pagamento de uma parcela. O valor original dela é recebido via parâmetro e o método deve abatê-lo do saldo devedor.

+ *exibeDados* – Mostra na tela todos os valores de todos os atributos, com os devidos títulos.

+ métodos *get's e set's* que o grupo achar necessários.

Classe **Compra**

Atributos privados (somente estes):
- modalidade – inteiro que indica uma das seguintes modalidades de compra:
  1 – à vista
  2 – parcelada, com 50% de entrada e o restante em duas parcelas iguais
  3 – parcelada em três parcelas iguais, sem nenhuma entrada.
- cliente – objeto do tipo **Cliente** que indica quem faz a compra
- data – objeto do tipo **Data** que indica a data em que a compra é realizada
- preço – valor original da compra, sem qualquer abatimento
- preço final – é o preço, abatido algum desconto promocional
- p1, p2, p3 – três atributos do tipo **Parcela**, que correspondem, respectivamente, à primeira, segunda e terceira parcela, se houver.

Construtor 1: com três parâmetros: cliente que faz a compra (tipo **Cliente**), data da compra (tipo **Data**) e preço. Atualiza os atributos e chama o método *escolheModalidade* para registrar a modalidade.

Construtor 2: faz o mesmo que o construtor 1, com a diferença com respeito à data da compra. Em lugar de um parâmetro de tipo objeto, receberá três parâmetros inteiros, com dia, mês e ano em que a compra está sendo feita. Com esses três dados, o construtor deve instanciar um objeto **Data** para atribuir ao respectivo atributo.

Métodos:
+ *escolheModalidade* – Exibe na tela do usuário as três modalidades para que o cliente escolha uma delas, o que é feito por ele digitando um inteiro no teclado. Lembrando, as modalidades podem ser 1, 2 ou 3. Se o usuário digitar um valor diferente, assumir o valor 1. Atualizar o atributo *modalidade.*

- *ultimasCrescente* – método privado que recebe via parâmetro o valor da última compra e retorna true ou false indicando se as três últimas compras estão em ordem crescente ou não.

+ finaliza a compra – este é o método mais importante do trabalho e valerá mais pontos. Vai calcular o preco final da compra, conforme as regras da empresa para as várias modalidades, vai instanciar as devidas parcelas, no caso de compra parcelada, e atualizar vários atributos.
Regras para o cálculo de descontos:
Modalidade 1 – à vista - aplica uma das três situações abaixo (só uma delas):

- desconto de 20% sobre o preço, se o mês da compra coincide com o mês de aniversário do cliente;
- 8% de desconto, se as três últimas compras, incluindo esta que está sendo feita, formam uma sequência crescente; esta verificação deve ser feita chamando o método *ultimasCrescente*;
- 5% de desconto, se for apenas compra à vista.

Modalidade 2 – parcelada, com 50% de entrada e duas parcelas iguais. Recebe desconto de 3,5% sobre o preço.
Modalidade 3 – parcelada em 3 vezes iguais. Não ganha desconto.

O método deve instanciar as parcelas, se houver, e atualizar os atributos preço final, p1, p2 e p3, desta classe. Para calcular a data de vencimento de cada parcela, chamar o método *calculaVencimentoParcela.* Também deve, conforme a situação, atualizar atributos da classe **Cliente*,*** chamando os métodos adequados. Ao final, o método deve retornar um string contendo um dos seguintes textos, conforme o caso:

Compra à vista, ganhou 20% de desconto, pois cliente nasceu em xx/xx/xxxx.
Compra à vista, ganhou 8% de desconto.
Compra à vista, ganhou só 5% de desconto.
Compra com entrada + 2 parcelas, ganhou desconto de 3,5%.
Compra em 3 parcelas, não ganhou desconto.

+ *calculaVencimentoParcela* – a regra para determinar a data de vencimento de cada parcela é a seguinte: o vencimento da primeira parcela é sempre no dia 28 do mês imediatamente seguinte ao mês da compra; cada parcela subsequente vai vencer no dia 28 do mês seguinte ao da parcela anterior. Por exemplo, se uma compra parcelada em 3 vezes é feita no dia 12/11/2016, as três parcelas vão vencer, respectivamente, em 28/12/2016, 28/1/2017 e 28/2/2017.
Este método recebe uma data (objeto) e instancia e retorna um objeto **Data** que corresponde ao dia 28 do mês imediatamente seguinte.

+ exibeDados - imprime os valores dos atributos, com títulos e organização agradáveis à leitura. Chamar métodos de saída de outras classes, quando for o caso. Exibir detalhes das parcelas, só para as que realmente existem. Por exemplo, se a compra é à vista, não deve aparecer nada sobre parcelas.

+ métodos *get* e *set* que o grupo julgar necessários para o funcionamento ou teste da aplicação.

Classe **TestaCompras**

Completar a programação do método *main* de acordo com o que pede cada comentário:

```
/** Alunos: xxxxxxxxx e xxxxxxxxx    Trabalho GA Lab 1    turma: xx  2016/2*/

public class TestaCompras{
  public static void main(String[] args){

    // Instanciar um cliente, para o qual serão instanciadas as 5 compras abaixo


    // Instanciar uma compra à vista (usar o primeiro construtor), com mês da
    // compra igual ao do aniversário do cliente, finalizar a compra e exibir
    // dados da compra.


    // Instanciar para o mesmo cliente uma segunda compra (usar o segundo
    // construtor), de valor mais alto que a primeira, para pagamento com
    // entrada e duas parcelas, finalizar a compra e exibir dados dela.


    // Instancia uma nova compra à vista, num mês diferente do de seu
    // aniversário e com valor maior que os das compras anteriores. Finalizar e
    // exibir dados da compra.


    // Instanciar uma compra à vista, fora do mês de aniversário, com valor
    // menor que o da compra anterior. Finalizar e exibir dados da compra.


    // Instanciar uma compra em 3 parcelas. Finalizar e exibir dados dela.


    }
}
```

Evite cedilhas e acentuação no código Java.

Entrega, conforme as datas do Plano de Ensino:

- Em papel: os diagramas UML das classes que aparecem no início do enunciado e um diagrama de objetos de uma compra na modalidade 2, de um cliente com apenas uma compra anterior. Invente dados adequados.

- Postar na comunidade: as classes em Java. Não esquecer o seguinte comentário no início do texto fonte de cada classe:
/ Alunos : xxxxxxxxxxxxx e xxxxxxxxxxx Trabalho GA Lab 1 turma: XX 2016/2 */**