1a) Construa uma classe `CartaoTelefonico` contendo como atributo a quantidade de créditos disponíveis pelo cartão. Implemente nesta classe um método construtor que receba como parâmetro a quantidade de créditos inicial do cartão, e os métodos de acesso/modificadores para o atributo da quantidade de créditos. Crie um método `status` que imprima a quantidade de créditos do cartão.

1b) Construa uma classe `Orelhao` contendo como atributo um número inteiro representando a área DDD (exemplo: 11, 21,...) em que está localizado o orelhão e a quantidade de ligações realizadas pelo aparelho. Implemente nesta classe um método construtor que receba como parâmetro a área em que o orelhão está localizado. Todo objeto instanciado da classe `Orelhao` deve estar com a quantidade de ligações zeradas. Crie um método `fazLigacao(int area, CartaoTelefonico cartao, int tempo)` que receba como parâmetro a área do número do telefone a ser ligado, um objeto da classe `CartaoTelefonico` , e o tempo em segundos da ligação. Este método deve ser responsável de incrementar a quantidade de ligações e atualizar o crédito do cartão. Sabe-se que uma ligação local consome 1 crédito a cada 20 segundos, e uma ligação fora da localidade, consome 1 crédito a cada 10 segundos. Ligações só devem ser realizadas para cartões com créditos suficientes para satisfazer o tempo indicado. Crie um método `status` que imprima a área DDD e o número de ligações realizadas.

```
public class Prog {
        public static void main(String args[])
        {
                CartaoTelefonico c1, c2;
                Orelhao o1;
                c1 = new CartaoTelefonico(20);
                c2 = new CartaoTelefonico(30);
                o1 = new Orelhao(11);

                o1.fazLigacao(11,c1,120);
                o1.fazLigacao(21,c1,100);
                o1.fazLigacao(42,c1,120);
                o1.fazLigacao(31,c2,120);
                o1.fazLigacao(21,c2,60);
                o1.fazLigacao(11,c2,80);

                c1.status();
                c2.status();
                o1.status();
        }
}
```

2a) Construa uma classe `Piloto` contendo como atributos a quantidade de horas de vôo e o nome do piloto. Implemente nesta classe um método construtor que receba como parâmetro uma string com o nome do piloto, e os métodos de acesso/ modificadores apenas para o atributo do nome do piloto. Todo objeto piloto instanciado deve inicialmente estar com a quantidade de horas de vôo zerada. A classe deve disponibilizar um método `adicionaHoras(int horas)` que adiciona uma quantidade de horas de vôo ao piloto. Crie um método `imprime` que imprima o nome do piloto e a quantidade de horas de vôo, sendo que deverá ser impresso o tipo do piloto: até 200 horas, *co-piloto*, caso contrário *comandante*.

2b) Construa uma classe `Aviao` contendo como atributo um número inteiro representando a quantidade de horas de atividade do avião e implemente nesta classe um método construtor que inicialize este atributo com zero. Crie um método `fazVoo(int horas, Piloto piloto)` que receba como parâmetro a quantidade de horas do vôo, e um objeto da classe `Piloto`. Este método deve ser responsável de incrementar a quantidade de horas de atividade do avião e atualizar a quantidade de horas de vôo do piloto. Sabe-se que um avião faz uma revisão a cada 200 horas de atividade. Vôos só devem ser realizados para aviões que tenham feito revisões a cada 200 horas. Crie um método `fazRevisao()` que zera a quantidade de horas de atividade do avião e um método `status()` que imprima a quantidade de horas de atividade e se o avião precisa ou não de revisão.

```
public class Prog {
        public static void main(String args[])
        {
                Piloto p1, p2;
                Aviao a1;
                p1 = new Piloto("Abreu");
                p2 = new Piloto("Julio");
                a1 = new Aviao();

                a1.fazVoo(40,p1);
                a1.fazVoo(100,p2);
                a1.fazVoo(170,p2);
                a1.fazVoo(30,p1);
                a1.fazRevisao();
                a1.fazVoo(20,p1);

                p1.imprime();
                p2.imprime();
                a1.status();
        }
}
```

3a) Construa uma classe `BilheteEvento` contendo um atributo inteiro para controlar se o bilhete já foi usado para entrada (valor 1) e saída (valor 2) em um evento . Implemente nesta classe um método construtor que inicialize este atributo com zero, e um método de acesso que retorne o valor deste atributo. A classe deve disponibilizar um método `entrada()`, e um método `saida()`, para registrar a entrada e saída. Crie um método `imprime` que imprima qual a situação do bilhete.

3b) Construa uma classe `CatracaEvento` contendo como atributo um número inteiro representando a quantidade de pessoas presentes no evento e implemente nesta classe um método construtor que inicialize este atributo com zero. Crie um método `registraEntrada(BilheteEvento bilhete)` e um método `registraSaida(BilheteEvento bilhete)` ambos recebendo como parâmetro um objeto da classe `BilheteEvento`. Estes métodos devem ser responsáveis de registrar a entrada e saída dos bilhetes e atualizar a quantidade de pessoas presentes no evento. Sabe-se que um bilhete com entrada já registrada não pode realizar novas entradas, como também bilhetes com registro de saída não podem sair e nem entrar novamente. A quantidade de pessoas presentes no evento é contabilizada pelo número de bilhetes que deram entrada e que não registraram saída. Crie um método `imprime` que imprima a quantidade de pessoas no evento.

```
public class Prog {
        public static void main(String args[])
        {
                BilheteEvento b1, b2, b3;
                CatracaEvento c1;
                b1 = new BilheteEvento();
                b2 = new BilheteEvento();
                b3 = new BilheteEvento();
                c1 = new CatracaEvento();

                c1.registraEntrada(b1);
                c1.registraEntrada(b2);
                c1.registraEntrada(b4);
                c1.registraSaida(b3);
                c1.registraSaida(b2);
                c1.registraEntrada(b3);
                c1.registraSaida(b2);

                c1.imprime();
                b1.imprime();
                b2.imprime();
                b3.imprime();
        }
}
```

4a) Construa uma classe `TimeFutebol` contendo como atributos: o nome do time, um inteiro que representa a quantidade de jogos realizados pelo time, um inteiro que indica o número de pontos ganhos, e um inteiro que segue a seguinte representação: 0 – retranqueiro, 1 – joga no ataque. Implemente nesta classe um método construtor que receba como parâmetro uma string com o nome do time e um inteiro indicando se o time é retranqueiro ou joga no ataque. Este método construtor deve inicializar a quantidade de jogos e os pontos ganhos com zero. Crie apenas um método de acesso para retornar o valor do estilo de jogo do time. A classe deve disponibilizar um método `venceu()`, um método `empatou()`, e um método `perdeu()` para registrar respectivamente a vitória (3 pontos), empate (1 ponto), e derrota (0 pontos), atualizando devidamente os pontos ganhos e jogos realizados. Crie um método `imprime` que imprima qual a situação do time.

4b) Construa uma classe `Campeonato` contendo como atributo um número inteiro representando a quantidade de jogos realizados e implemente nesta classe um método construtor que inicialize este atributo com zero. Crie um método `realizaJogo(TimeFutebol timeCasa, TimeFutebol timeVisitante)` que recebe como parâmetro dois objeto da classe `TimeFutebol`. Este método é responsável em analisar e registrar o resultado do jogo para cada time. Sabe-se que um time retranqueiro sempre ganha quando joga na casa de um adversário que joga no ataque. Times que jogam no ataque em jogos na casa do adversário sempre ganham. As outras combinações, sempre resultam em empate. Crie um método `imprime()` que imprima a quantidade de jogos realizados no campeonato.

```
public class Prog {
        public static void main(String args[])
        {
                TimeFutebol t1, t2, t3;
                Campeonato c1;
                t1 = new TimeFutebol("Bimboca",0);
                t2 = new TimeFutebol("BocaSeniors",0);
                t3 = new TimeFutebol("TerceiroTDS",1);
                c1 = new Campeonato();

                c1.realizaJogo(t1,t2);
                c1.realizaJogo(t2,t3);
                c1.realizaJogo(t3,t1);
                c1.realizaJogo(t3,t2);

                c1.imprime();
                t1.imprime();
                t2.imprime();
                t3.imprime();
        }
}
```