1) Programe as classes **Avaliacao** e **Aluno**.
A classe **Avaliacao** deve declarar dois atributos para armazenar a nota do grau A e a nota do grau B. A classe Avaliacao deve ter um construtor que recebe as notas, via parâmetros. As funcionalidades da classe são oferecidas pelos seguintes métodos:
setNotaGrauA, ***setNotaGrauB***, ***getNotaGrauA***, ***getNotaGrauB*** *e* ***calculaMedia*** que calcula a média ponderada com peso 2 para o grau B e peso 1 para o grau A.

A classe **Aluno** deve declarar dois atributos para armazenar o nome do aluno e a avaliação. A classe deve oferecer três construtores:
a) oferece a possibilidade de criar um aluno somente com o nome. Recebe apenas o nome como parâmetro.
b) oferece a possibilidade de criar um aluno com o nome e a avaliação. Recebe, via parâmetros, o nome do aluno e o objeto do tipo Avaliacao.
c) oferece a possibilidade de criar um aluno com o nome e a avaliação. Recebe, via parâmetros, o nome do aluno e as duas notas.
As funcionalidades da classe Aluno são oferecidas pelos seguintes métodos:
setNome, ***getNome***, ***setAvaliacao****(recebe duas notas, via parâmetros)*, ***getAvaliacao***

2) Escreva uma classe **TestaAluno**, com as seguintes operações, chamando os métodos adequados:

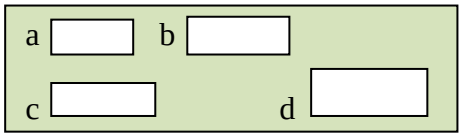
a) Instanciar (criar) um objeto Aluno com o nome e as duas notas lidas do teclado;
b) Instanciar outro objeto Aluno, somente com nome (lido do teclado);
c) Ler do teclado as duas notas do segundo aluno, e setar a avaliação deste aluno com estas notas.
d) Ler do teclado duas notas de um terceiro aluno. Instanciar um objeto Avaliacao, com estas notas. Instanciar um terceiro aluno com o seu nome e o objeto Avaliacao instanciado.
e) Calcular as médias dos três alunos e armazenar em variáveis adequadas;
f) Exibir na tela o seguinte relatório:

Relatório de notas			
Nome do aluno	Notas		Média
xxxxxxxxxxxx	xx.xx	xx.xx	xx.xx
xxxxxxxxxxxx	xx.xx	xx.xx	xx.xx
xxxxxxxxxxxx	xx.xx	xx.xx	xx.xx

3) Crie a classe **TestandoTrocaDeVariaveis**, sem atributos, contendo somente o método main, o qual deverá realizar as operações abaixo especificadas. Desenhe a memória e mostre o que está ocorrendo quando cada instrução é executada:

a) Declare e inicialize quatro variáveis locais inteiras com as instruções:
int a = 65; int b = 10; int c = -2; int d = 33;

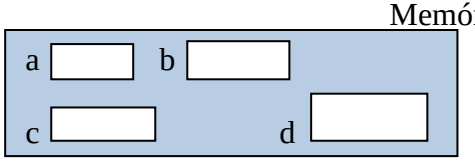
Memória



b) Exiba os valores das quatro variáveis com o cabeçalho "VALORES INICIAIS"

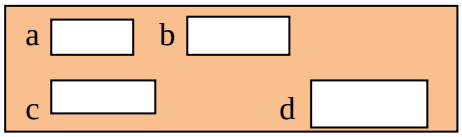
c) Trocar os conteúdos de **a** e **b** e exibi-los na tela com o cabeçalho "VALORES APÓS A TROCA DE a E b";

Memória



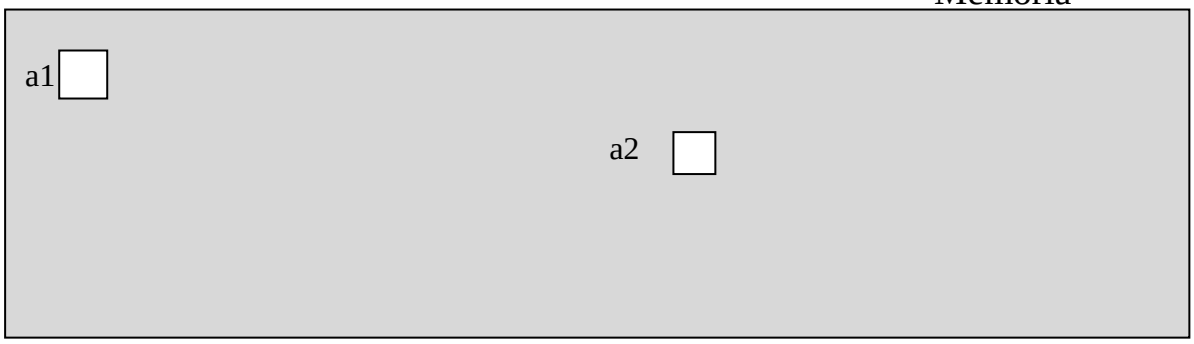
d) Trocar os conteúdos das quatro variáveis assim: **a** vai para **b,** **b** vai para **c,** **c** vai para **d** , **d** vai para **a**, e exiba os valores das quatro variáveis, com cabeçalho adequado;

Memória



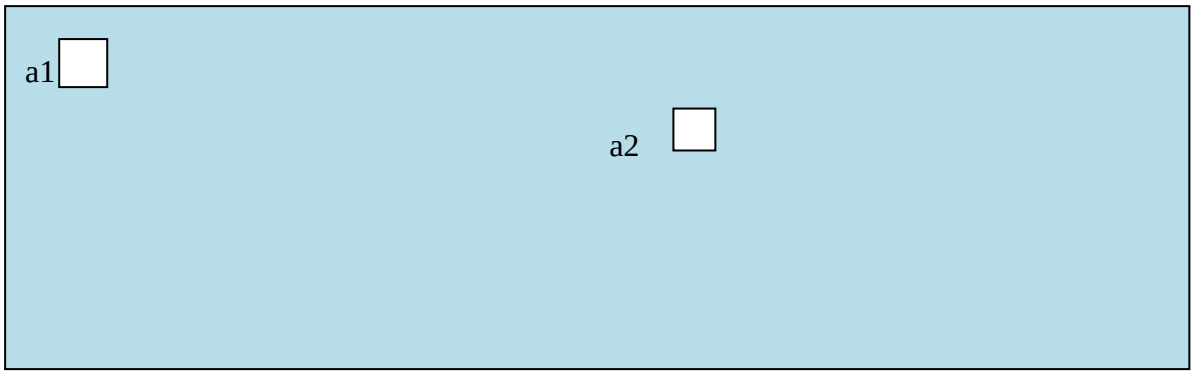
e) Instanciar dois objetos do tipo **Aluno** nas variáveis objeto a1 e a2 com os nomes "Maria" e "Pedro" e as notas lidas do teclado;

Memória



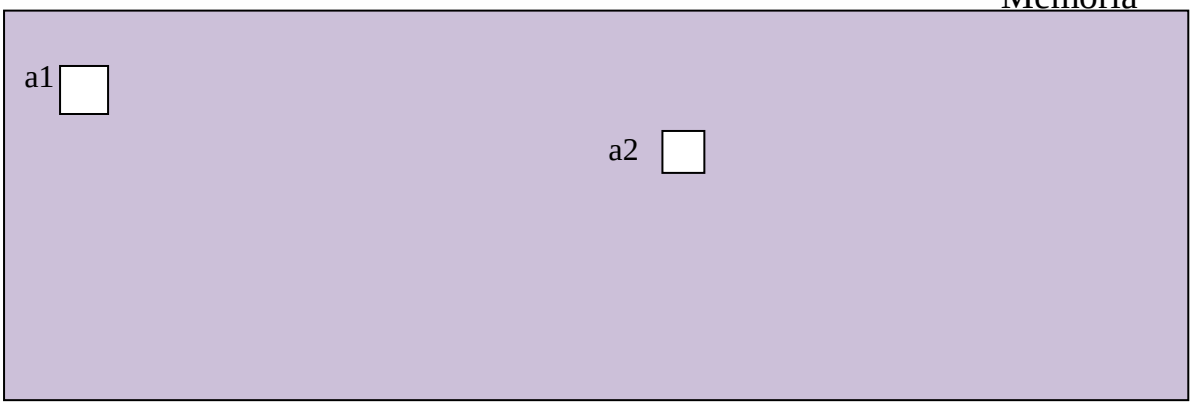
f) Trocar, entre si, as variáveis **a1** e **a2**, e imprimir os nomes dos alunos, obtidos pela chamada do método get, para conferir;

Memória



g) Escrever o comando a1 = a2; e depois exibir os nomes de ambos os alunos, obtidos pela chamada do método get, e explicar o que aconteceu.

Memória



h) Faça outros testes para resolver suas dúvidas e/ou curiosidades.

Atenção! os exercícios abaixo devem ser resolvidos após a aula de prog I, na qual será visto o comando de seleção (if).

4. Modifique os métodos *setNotaGrauA* e *setNotaGrauB* da classe **Avaliacao** (exercício 1) para não permitir que valores inválidos sejam armazenados nos atributos. As notas válidas são de 0 a 10.

5. Modifique a classe **VentiladorAnalogico** de uma lista anterior, não permitindo que o objeto fique em um estado inconsistente.

6. Implemente a classe **VentiladorDigital**. Considere um ventilador com 4 botões de velocidade (0, 1, 2 ou 3). A classe deve oferecer dois construtores: um que cria um

ventilador desligado (na velocidade 0) e outro que cria um ventilador em qualquer velocidade válida. Os seguintes métodos devem ser oferecidos:

- *getVelocidade*: retorna a velocidade atual
- *liga* : se o ventilador não está ligado, coloca-o na velocidade 1.
- *desliga*: coloca o ventilador na velocidade 0.
- *trocaVelocidade*: muda a velocidade do ventilador

7. a) Programe a classe **TresNumerosInteiros** com 3 atributos inteiros positivos (primeiro, segundo e terceiro). A classe deve oferecer um construtor com os 3 parâmetros e os seguintes métodos:

- **setPrimeiroNumero**: configura o primeiro atributo. Aceita somente positivo. Se valor inválido, atribui 1.
- **setSegundoNumero**: configura o segundo atributo. Aceita somente positivo. Se valor inválido, atribui 1.
- **setTerceiroNumero**: configura o terceiro atributo. Aceita somente positivo. Se valor inválido, atribui 1.
- **getPrimeiroNumero**: devolve o primeiro atributo.
- **getSegundoNumero**: devolve o segundo atributo.
- **getTerceiroNumero**: devolve o terceiro atributo.
- **getMaior**: devolve o maior número. Se iguais, devolve qualquer um.
  obs. o método deve executar o menor número possível de testes.
- **getMenor**: devolve o menor número. Se iguais, devolve qualquer um.
  obs. o método deve executar o menor número possível de testes
- **classificaOrdemCrescente**: classifica os 3 números em ordem crescente. Os atributos do objeto devem ficar em ordem crescente.

7. b) Faça uma classe de teste:
i) ler 3 números inteiros positivos do teclado com a seguinte tela de entrada:
Obs. se o número digitado não for positivo, exibir a mensagem "Primeiro NUMERO INVALIDO " , ou segundo ou terceiro e encerrar o programa.
Atenção! Se um dos números digitados é inválido, os outros números não devem ser lidos.

```
      ENTRADA DE DADOS:
Digite o PRIMEIRO NUMERO:

Digite o SEGUNDO NUMERO:

Digite o TERCEIRO NUMERO:
```

ii) criar um objeto da classe **TresNumerosInteiros** com os números lidos do teclado.
iii) exibir o maior e o menor, chamando os métodos adequados.
iv) exibir quantos dígitos tem o maior e quantos dígitos tem o menor. Converta para String. (consulte a minibiblioteca)
v) se o menor número tiver menos de 3 dígitos, exibi-lo com três dígitos (com zeros à esquerda).
vi) alterar o segundo atributo para um valor aleatório (use o método random).
vii) chamar o método para classificar os atributos em ordem crescente e exibi-los após a classificação.
viii) criar outro objeto da classe **TresNumerosInteiros** com números inteiros gerados pelo método *random* da classe **Math**. Gere números no intervalo [5,105).
ix) alterar o segundo número, para que este fique igual à diferença do primeiro pelo segundo, chamando o método *setSegundoNumero*. Exibir os 3 números.
x) trocar, entre si, os valores do terceiro atributo dos dois objetos, chamando os métodos adequados.
xi) exibir os valores dos atributos dos dois objetos.
xii) exibir o maior valor (entre os 6 valores), chamando os métodos adequados.

8. Implemente a classe **Carro** com atributos para armazenar a quantidade de combustível no tanque, o consumo e a quilometragem percorrida.
Faça dois construtores: um que cria um carro zero de tanque vazio que faz 10 Km por litro e outro que recebe a quantidade de combustível inicial, a quilometragem e o consumo. Sabe-se que o tanque do carro suporta no máximo 55 litros.

Faça os métodos:
-*setConsumo* – configura o atributo consumo com o valor recebido como parâmetro. O consumo não pode ser zero nem negativo. Se receber um atributo inválido, setar o consumo com o valor default (10 Km por litro);

-*getConsumo* – retorna o valor do atributo consumo;

-*abastece* – recebe como parâmetro a quantidade de combustível para abastecer o carro, verifica se há espaço no tanque e devolve true ou false, indicando se o carro foi abastecido ou não. Atualiza adequadamente o atributo;

-*anda* - recebe como parâmetro a quantidade de quilômetros a serem percorridos, verifica se há combustível suficiente e devolve true ou false, indicando se o carro andou ou não. Atualiza os atributos adequadamente;

-*status* – devolve um String com os dados do carro.