*Olá Caros alunos, segue abaixo a relação entre os comandos Java e Portugol*

*que vocês irão preencher apenas os que faltam.*

*Após esta relação temos as questões de Java que iremos trabalhar,*

*Cada exercício está separado pelos* ***Conceitos*** *que trabalhamos em sala.*

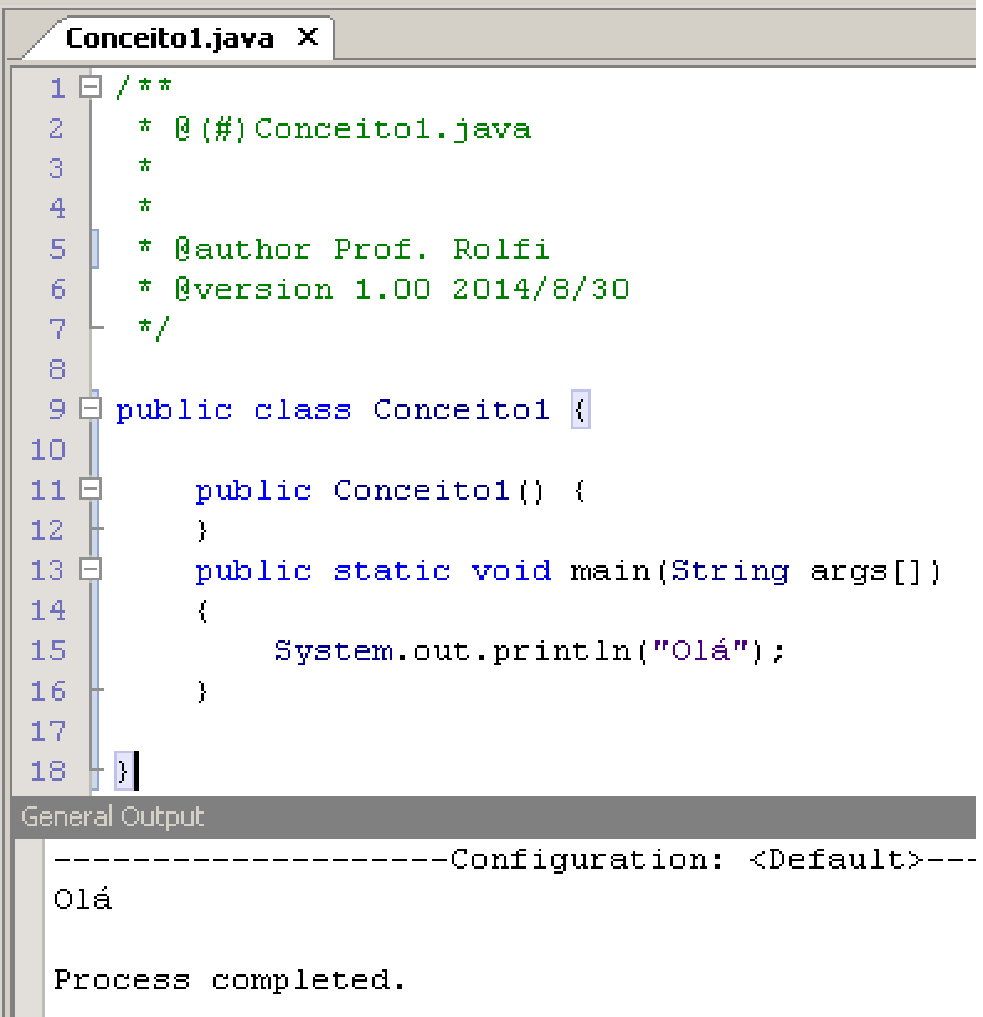
*Irei mostrar* ***um exemplo*** *e algumas* ***explicações*** *e suas respectivas* ***questões****.*

*Bons estudos !!!!*

|  |  |
| --- | --- |
| **Complete os comandos que faltam com Portugol**  OBS: As questões que tiverem <comando1>;<comandos>;< comandos a serem repetidos >; considerem tal como está | |
| Comando em Portugol | Comando em Java |
| escreva | System.out.print |
| escreval | System.out.println |
| var  numero:inteiro  peso:real  nome: caracter | int numero;  float peso;  String nome; |
| escreva(“Seu nome é:”**,** nome) | System.out.println(“Seu nome é:”**+**nome); |
| algoritmo “Aula1”  var  inicio  fimalgoritmo | class Aula1{  public static void main(String args[])  {  }  } |
|  | **Expressão Aritmética:**  double num1 = 1 + (2 - num3 \* (num2/num4));  int num=0;  num++; |
|  | **Expressão lógica:**  boolean valor= num1 > num3; |
| se <expressão lógica > entao  <comando1>  <comando2>  fim se | if ( expressão lógica) {  <comando 1>;  <comando 2>;  } |
| se <expressão lógica > entao  <comando1>;  <comando2>;  senão  <comando3>;  <comando4>;  fim se | If ( expressão lógica){  <comando1;>  <comando2;>  }else{  <comando3;>  <comando4;>  } |
| se <expressão lógica 1 > entao  <comando1>;  senão se <expressão lógica 2> entao  <comando2>;  senão  <comando3>;  fim se  fim se | If ( expressão lógica){  <comando1;>    }else if( expressão lógica){  <comando2;>  }else{  <comando3;>  } |
| numero<-4  escolha( numero)  caso 0: <comandos> ; sair;  caso 1: <comandos> ; sair;  caso 2: <comandos> ; sair;  caso 3: <comandos> ; sair;  caso 4: <comandos> ; sair;  outrocaso: // Caso nenhum seja certo  fimescolha | int numero=4;  switch( numero)  {  case 0:<comandos> ; break;  case 1: <comandos>; break;  case 2: <comandos>; break;  case 3: <comandos>; break;  case 4: <comandos>; break;  default: // Caso nenhum seja certo  } |
|  | for(int contador=0; contador< 10; contador++) {  <comandos a serem repetidos>;  System.out.println(“Número:”+contador);  } |
|  | int contador=0;  while(contador<10)  {  <comandos a serem repetidos>;  System.out.println(“Número:”+contador);  contador++;  } |
|  | int contador=0;  do{  System.out.println(“Número:”+contador);  <comandos a serem repetidos>;  contador++;  }while(contador<10); |
|  | String cadeiaNomes[]=new String[10]; |
|  | String mundo[][]=new String[3][3]; |

|  |
| --- |
| **Conceito 1: Mostrar valores aos usuários** |

Exemplo I- Faça um programa em Java que mostre na tela "Olá", leia o código-fonte abaixo:



Explicação:

Pessoal, da linha 1 até a linha 7 do código são os **comentários**, lembram quando o código fica "verde" ? Então significa que o compilador não irá rodar. Temos vários comentários o **//** - que comenta a linha inteira e o /\* que comenta várias linha juntas \***/,** observe acima que usamos o **/\*\*/**.

O código mesmo inicia-se da linha 9, inicia-se com a linha **public class** que é obrigatório para qualquer programa feito em Java. Após o public class vem o nome do **programa**, no Java um programa é chamado de **Objeto** ou **Classe**, neste caso nossa Classe tem o nome de **Conceito1;** a classe poderia ser qualquer nome, desde que inicie o (a) nome com letra e (b) não contenha espaços em brancos.

Na linha 11 nós temos o **Construtor**, bem uma metáfora de construtor, seria um exemplo de loading de programas de celulares ou video-games, no construtor são iniciados alguns objetos e valores em memória que a Classe precisa, no nosso caso acima não contém nada no código, bem por enquanto não é obrigatório tê-lo.O Construtor tem o nome da Classe que é definido no public class, que por sua vez é o mesmo nome do arquivo.

Na linha 13 temos o **public static void main (String args[])** que é obrigatório, apenas a palavra **args**[] poderia ser mudado dentre esta linha de código, no entanto por definição deixamos sempre assim. Esta linha de código é o corpo do programa onde roda as linhas de códigos da Classe.

Na linha 14 temos a abertura **{** e na linha 16 temos o fechamento **}** do **main,** assim como no final da linha 9 **{** e no final da Classe na linha 18 **}**. No Java a abertura e o fechamento é o que determina um **bloco.** Ao Iniciar uma classe é aberto um bloco, e dentro deste Bloco é escrito o Bloco do **main**.



Na linha 15 temos a impressão na tela de **"Olá"**, no comando **System.out.println("Olá");** este comando imprime para o usuário e tudo que é escrito dentro das **"** (aspas duplas) **"** , é impresso na tela tal como está escrito.

OBS: Todos os códigos e instruções em Java terminam com o delimitador **;** (ponto e vírgula), exceto alguns poucos códigos.

Questões do Conceito 1:

1- Mostre ao usuário a Palavra "Teste", o nome da Classe será ExercicioTeste, deve ter construtor.

2- Mostre ao usuário a Frase " 1º Primeiro teste do Java !!!! ", o nome da Classe será ExercicioFrase, não precisa de construtor.

|  |
| --- |
| **Conceito 2 : Variáveis** |

Exemplo II: Faça uma Classe em Java que imprima o nome, idade, peso e sexo de um cliente **pré-determinado**.

Explicação (Leia o código-fonte nas páginas seguintes): No caso deste programa nós usamos as variáveis já com valores que colocamos no caso "Pedro A. Cabral", é o nosso cliente. Quando falamos em valores **pré-determinados** não pedimos ao usuário que defina algum valor.

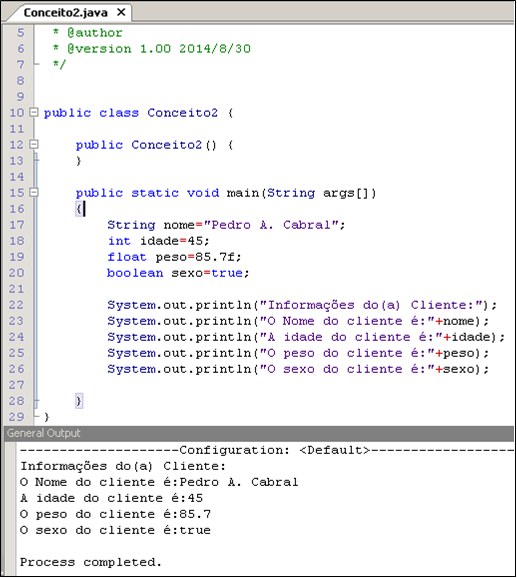
O que é Variável ? São espaços na memória que definimos o Tipo, A identificação e O Valor.

No Java os **Tipos** mais comuns são :

|  |
| --- |
| **String** – Valores com Letras , Caracteres e Números. |
| **int** – Valores sem ponto ou vírgula, não números inteiros. |
| **float** – Valores com ponto e vírgulas, são números "quebrados" ou flutuantes, sendo obrigatório o uso do **f** após o número. |
| **boolean** – Apenas **true** ou **false**, verdadeiro ou falso. |
| **double** – como se fosse um float com maior espaço em memória. |
| **longint** – como se fosse um int com maior espaço em memória. |

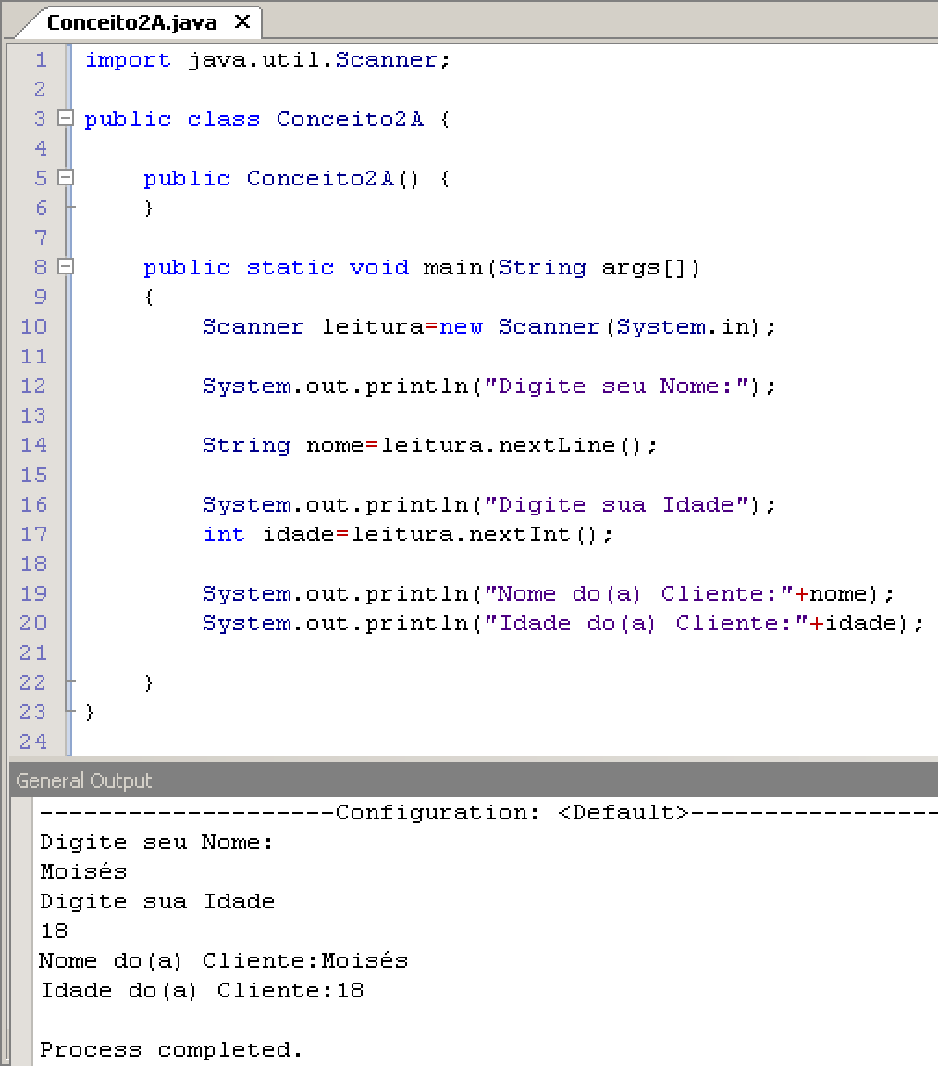
A **identificação** e o **valor** fica a critério do programador.

Na linha 22, o comando **System.out.println("O Nome do cliente é:" + nome);**



Questões Conceito 2:

3-Criando uma classe com nome CadastroDeClientesPre, deverá imprimir : nome, data de nascimento, rg, cpf, telefone, email, endereço completo, sem construtor com dados pré-determinados.



|  |
| --- |
| **Pegar valores dos Usuários** |

Exemplo IIA: Faça uma Classe em Java que solicite ao usuário o nome e idade de um cliente e imprima seus dados na tela.

Explicação IIA: Deveremos usar a classe **Scanner**, através do comando **import java.util.Scanner;** este comando importa a classe do pacote Java.Util a Classe Scanner. A classe Scanner serve para pegarmos valores do teclado.Todo comando de **import** , deve ser digitado antes de abrir a Classe no caso do exemplo abaixo está na linha1 e logo após na linha2 abre-se a **classe** **Conceito2A**. Na linha 1 "importarmos uma biblioteca".

Dentro do Bloco do Main, na linha 10 definimos a Classe **Scanner** na variável chamada **leitura,** como o comando :

**Scanner leitura=new Scanner(System.in);**

**L**ogo toda vez que chamarmos **leitura** será como se tivessemos chamando a própria **Scanner**. Na linha 10 "Escolhemos 1 livro (Scanner) na 1º página (System.in)".

Na linha 14 para pegarmos valores sendo letras ou números usamos o **leitura.nextLine()** para variáves **String**.

Na linha 17 usamos o **leitura.nextInt()** para pegarmos valores para variáveis **int**.Se fosse para pegar valores flutuantes como peso , dinheiro, altura , usariamos **float peso=leitura.nextFloat();**

**Exercício**

4-Criando uma classe com nome CadastroDeClientesInfo, deverá **solicitar** ao usuário que digite os dados : nome, data de nascimento, rg, cpf, telefone, email, endereço completo, com construtor e após ele digitar imprimir todos os dados na tela.

|  |
| --- |
| **Conceito 3: Expressões Lógicas** |

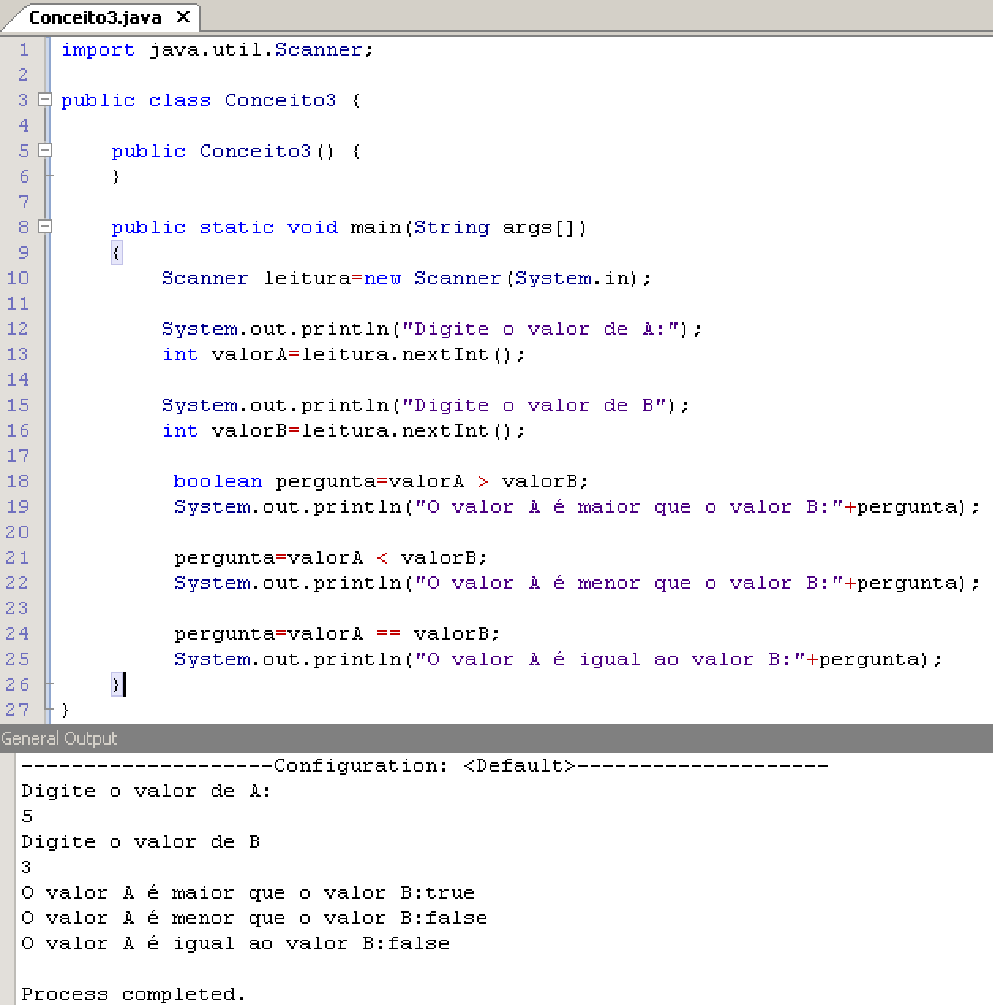
Exemplo III: Faça um programa que receba dois números e retorne se o primeiro número é maior , menor ou igual, ao segundo número através das expressões lógicas.

Explicação: As expressões lógicas são "contas" que retornam **true** ou **false**, é como se fosse uma pergunta, na linha18 , é criado uma variável **boolean** chamada **pergunta**. E nessa variavel é gravado a expressão ou resultado da pergunta, A é maior que B, em Java, **pergunta=valorA > valorB;** caso o usuário digite 5 para **valorA** e 3 para **valorB** retorna **true** , pois 5 é maior que 3.

Assim sucessivamente para linha21, onde é "perguntado" se **valorA** é menor que **valorB**, em Java, **pergunta=valorA < valorB;** tendo como resultado **false** para os valores 5 e 3. E na linha 24 **pergunta= valorA == valorB;** é "perguntado" se estas duas variáveis são iguais, ou seja se valorA possui o mesmo valor para valorB, no caso do exemplo acima 5 e 3 a resposta da pergunta é **false**.

As **expressões lógicas** ("perguntas") são:

|  |  |
| --- | --- |
| Expressão Lógica | Exemplo |
| > - Maior | valorA > valorB |
| < - Menor | valorA < valorB |
| >= -Maior Igual | valorA >= valorB |
| <= - Menor Igual | valorA <= valorB |
| = = é Igual | valorA == valorB |
| != é Diferente | valorA != valorB |



Questões Conceito 3:

5- Resolva as expressões lógicas abaixo NO CADERNO:

|  |  |  |  |
| --- | --- | --- | --- |
| valorA | valorB | Expressão | Retorno |
| 5 | 7 | valorA > valorB | false |
| 7 | 25 | valorA < valorB |  |
| 4 | 4 | valorA >= valorB |  |
| 3 | 32 | valorA <= valorB |  |
| 5 | 5 | valorA == valorB |  |
| 5 | 5 | valorA != valorB |  |
| 5 | 6 | valorA != valorB |  |
| 5 | 6 | valorA != valorA |  |
| 5 |  | valorA != 3 |  |
| 5 | 3 | valorB != 3 |  |

|  |
| --- |
| **Portas Lógicas** |

Nesta mesma idéia também temos o "resultado das perguntas" uma metáfora de Portas lógicas, como assim ?

Exemplo:

**boolean resultado\_da\_expressao = (valorA < valorB) && (valorA==valorB) ;**

Para valorA=3 e valorB=5, essa expressão retorna false, pois a primeira "pergunta 1" (valorA < valorB) retorna um true, e a segunda "pergunta 2" (valorA == valorB) retorna false. Como a pergunta central é o **&&** logo todas as perguntas deveriam resultar em true para retorna true, mas como a "pergunta 2" retorna false o "resultado da pergunta" é false.

No caso de:

**boolean resultado\_da\_expressao = (valorA < valorB) || (valorA==valorB) ;**

Retorna true, pois na pergunta se tiver um que retorne true ja retorna para a variável **resultado\_da\_expressao**.

Após a pergunta do resultado\_da\_expressão, podemos imprimir :

**System.out.println("O resultado das expressões é:"+resultado\_da\_expressao);**

Como "resultado das perguntas" temos o nome técnico que é **Portas Lógicas**.

As portas lógicas mais comuns são:

|  |
| --- |
| && - Porta E : regra principal se tiver um false e todas true nas outras perguntas retorna false. |
| || - Porta OU: regra principal se tiver um true e todas false retorna true. |

**Exercício**

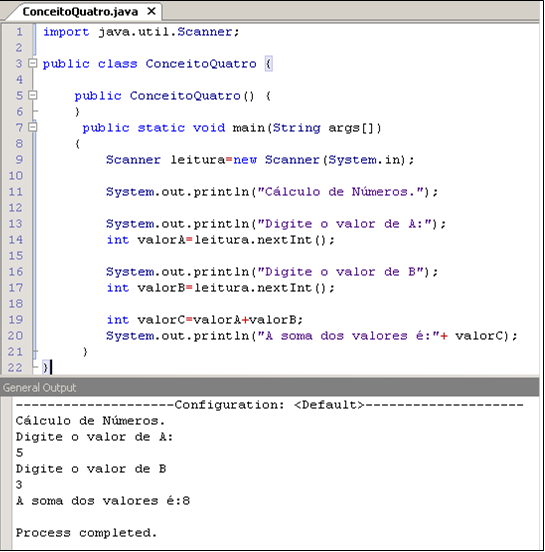
6- Resolva as expressões lógicas abaixo NO CADERNO:

|  |  |  |  |
| --- | --- | --- | --- |
| valorA | valorB | Expressão | Retorno |
| 5 | 7 | (valorA > valorB) && (valorA = = valorB) | false |
| 7 | 25 | (true) && (false) && (false) |  |
| 4 | 4 | (true) || (false) || (false) |  |
| 3 | 32 | (valorA > valorB) || (valorA = = valorB) |  |
| 5 | 5 | (valorA >= valorB) && (valorA = = valorB) |  |
| 5 | 5 | (valorA >= valorB) || (valorA = = valorB) |  |
| 5 | 6 | (valorA <= valorB) && (valorA = = valorB) |  |
| 5 | 6 | (valorA <= valorB) || (valorA = = valorB) |  |
| 5 | 5 | (true) && (valorA = = valorB) |  |
| 5 | 5 | (true) || (valorA = = valorB) |  |

7 - Faça uma Classe em Java que receba duas idades e dois pesos e dois nomes, e pergunte se são a mesma pessoa, colocando essas variáveis em uma expressão booleana onde todos devem retornar true se for iguais, imprimindo no System ... ("Usuários identicos : "+resultado\_da\_expressao); Esta classe retornará um true ou um false.

|  |
| --- |
| **Conceito 4 : Expressões Aritméticas** |

Exemplo IV:Faça um Objeto (Classe ou Programa) que peça 2 valores para o usuário e retorne a soma deles.



Explicação:Na linha 19 é criado uma variável chamada valorC, que soma os espaços em memória valorA e valorB que ambos são solicitados pelo usuário, após a soma é devolvido o valor para o usuário.No Java as "contas" são guardadas em variáveis e trabalhadas em tempo de execução do programa.

O Java suporta as operações básicas como : soma (**+**), subtração (**-**) , multiplicação (**\***) e divisão (**/**) , no caso da divisão o ideal é que a variável que recebe esta conta seja double ou float.

Também temos as funções **Math.sqrt(variavel ou número)** que retorna a raiz quadrada de uma variável ou número e a **Math.pow(a,b)** que é a função do elevado equivalente na matemática por **ab** .

E também o módulo **%** que é o resto da divisão de um número ou seja podemos verificar se um número é divisil por outro número.

Exemplo:

Conta (a):

10 / 2 = 5

5 \* 2 =10

10 – 10 = 0 - resto

Conta (b):

10 / 3 = 3,3333

3,3333 \* 3 = 9,9999

10 – 9,9999 = 0,0001 (arredonda para 1 )

Resultado: (A) 10 é divisivel por 2 e (B) 10 não é divisil por 3.

Questões Conceito 4:

8-Faça um programa que o usuário digite um ano e retorno o Resto da divisão por 4.

9-Faça um programa que o usuário digite um número e : some este número por 5 e imprima na tela, subtraia o valor do usuário por 7 e imprima na tela e faça o mesmo com a multiplicação por 10 e divisão por 2.