**Fundação valeparaibana de ensino**colégios univap – unidade centro

Curso técnico em informática

nome do autor (Individual)

.

LISTA DE EXERCÍCIOS 3º bimestre

PROGRAMAÇÃO AVANÇADA PARA WEB

Lista apresentada ao Curso Técnico de informática como composição de nota.

Prof. Me. Hélio Lourenço Esperidião Ferreira

SÃO JOSÉ DOS CAMPOS

2023

**PROGRAMAÇÃO AVANÇADA PARA WEB  
LISTA DE EXERCÍCIOS II – PHP BÁSICO**

**CONSTRUA APIS E UTILIZE ORIENTAÇÃO A OBJETOS**

**CASO O ALUNO NÃO UTILIZE OS PADRÕES DE ORIENTAÇÃO A OBJETOS, MVC E REST A LISTA SERÁ DESCONSIDERADA.**

1. Converter uma quantidade de horas digitadas pelo usuário em minutos. Informe o resultado em minutos. Crie a classe Horas (Horas.php)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Método | Rota | Envio de dados | Funcionalidade | Retorno |
| GET | /horas/minutos/*n* | /horas/minutos/5 | Transforma horas em minutos | { “horas”:5,  “minutos”:300  } |
| GET | /horas/segundos/n | /horas/segundos/5 | Transforma horas em segundos | { “horas”:5,  “segundos”:18000  } |
| GET | /horas/n | /horas/5 | Retorna horas em minutos e segundos | { “horas”:5,  “minutos”:300, “segundos”:18000  } |

1. Crie um programa que dada a idade de uma pessoa calcule quantos dias, horas, minutos e segundo essa pessoa já viveu

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Método | Rota | Envio de dados | Funcionalidade | Retorno |
| GET | /idades/dias/n | Idades/dias/1 | Retorna quantos dias equivalem a 1 ano | { “anos”:1,  “dias”:365  } |
| GET | /idades/horas/n | /idades/horas/1 | Retorna quantas horas equivalem a 1 ano | { “anos”:1,  “horas”:8760  } |
| GET | /idades/minutos/n | /idades/minutos/1 | Retorna quantos minutos equivalem a 1 ano | { “anos”:1,  “minutos”:525600  } |
| GET | /idades/segundos/n | /idades/segundos/1 | Retorna quantos segundos equivalem a 1 ano. | { “anos”:1,  “segundos”:31536000  } |
| GET | /idades/n | /idades/1 | Retorna quantos dias, horas, minutos e segundos equivalem a 1 ano. | { “anos”:1,  “dias”:365,  “horas”:8760,  “minutos”:525600,  “segundos”:31536000  } |

1. Crie um programa que leia duas notas de um aluno e apresente a média

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Método | Rota | Envio de dados | Funcionalidade | Retorno |
| GET | /alunos/nostas/n1/n2 | /alunos/nostas/7.5/6.1 | Retorna a média entre 7.5 e 6.1 | { “n1”:7.5,  “n2”:6.1  “media”:6,8,  “conceito”: “Aprovado”  } |
| GET | /alunos/nostas/medias | /alunos/nostas/7.5 | Retorna o conceito aprovado ou reprovado | { “media”:7.5, “conceito”: “Aprovado”  } |

1. Uma imobiliária vende terrenos retangulares. Faça um programa para ler as dimensões de um terreno e depois exibir a área e comprimento.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Método | Rota | Envio de dados | Funcionalidade | Retorno |
| GET | terrenos/base/altura | terrenos/10/20 | Retorna área | { “base”: 10,  “altura”:20,  “area”:200  } |
| GET | terrenos/ | {  “base”:10  “altura”:20  } | Retorna o conceito aprovado ou reprovado | { “base”: 10,  “altura”:20,  “area”:200  } |

1. Faça um programa que calcule e mostre valores de graus Celsius e graus Fahrenheit, cujos graus variem de n1 a n2 F de 1 em 1. A conversão de graus Fahrenheit (F) para graus Celsius (C) é dada por: C=5/9\*(F-32).

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Método | Rota | Envio de dados | Funcionalidade | Retorno |
| GET | /graus/n1/n2 | / graus /32/35 | Retorna graus ºC. | { “32ºF”:”0ºC”,  “33ºF”:”0,55ºC”, “34ºF”:”1,111ºC”,  “35ºF”:”1,666ºC”  } |
| GET | /graus/n1 | / graus /32 | Retorna o equivalente em ºC | { “32ºF”:”0ºC”  } |

1. Faça um programa que calcule e mostre a área de um trapézio. Sabe-se que: A = (base maior + base menor) \* altura) /2;

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Método | Rota | Envio de dados | Funcionalidade | Retorno |
| GET | /trapezios/b1/b2/h | / trapezios /3/5/2 | Calcula a área do trapézio | { “baseMaior”:”5”,  “BaseMenor”:”3”, “altura”:”2”,  “area”:”8”  } |
| GET | /trapezios/ | /trapezios  Envia um JSON  { “baseMaior”:”5”,  “BaseMenor”:”3”, “altura”:”2”  } | Recebe um json e retorna a área. | { “baseMaior”:”5”,  “BaseMenor”:”3”, “altura”:”2”,  “area”:”8”  } |

1. Calcular o salário líquido do funcionário sabendo que este é constituído pelo salário bruto mais o valor das horas extras subtraindo 8% de INSS do total. Serão lidos nesse problema o salário bruto, o valor das horas extras e o número de horas extras. Apresentar ao final o salário líquido.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Método | Rota | Envio de dados | Funcionalidade | Retorno |
| GET | /funcionarios/salarios | {  “salarioBruto”:1300,  “valorHoraExtra”:15,  “totalHorasExtras”:8  } | Apresentar ao final o salário líquido | {  “salarioBruto”:1300,  “valorHoraExtra”:15,  “totalHorasExtras”:8,  “salarioLiquido”:1306,40  } |

1. Efetuar a leitura do número de quilowatts consumido e calcular o valor a ser pago de energia elétrica, sabendo-se que o valor a pagar por quilowatt é de 0,12. Apresentar o valor total a ser pago pelo usuário acrescido de 18% de ICMS.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Método | Rota | Envio de dados | Funcionalidade | Retorno |
| GET | /quilowatts/n | /quilowatts/350 | Calcular o valor a ser pago | {  “quilowatts”:350,  “valorQuilowatts”:0,12,  “imposto”: 18%  “totalPagar”:  } |

1. Calcular a média de combustível gasto pelo usuário, sendo informado a quantidade de quilômetros rodados e a quantidade de combustível.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Método | Rota | Envio de dados | Funcionalidade | Retorno |
| GET | /combustiveis/medias | {  “kmRodados”:80,  “litrosCombustivel”:5  } | Calcular comsuno | {  “kmRodados”:80,  “litrosCombustivel”:5,  “consumo”: “16km/L”} |

1. Elabore um algoritmo que dada a idade de um nadador classifique-o em uma das seguintes categorias:

Infantil A = 5 a 7 anos

Infantil B = 8 a 11 anos

Juvenil A = 12 a 13 anos

Juvenil B = 14 a 17 anos

Adultos = Maiores de 18 anos

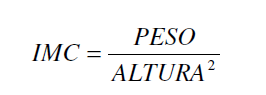
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Método | Rota | Envio de dados | Funcionalidade | Retorno |
| GET | /classificacoes/nadadores/idade | /classificações/nadadores/9 | Retorna classificação | {  “idade”:9,  “categoria”:Infantil B  } |

1. Elabore um algoritmo para calcular a equação do segundo grau e imprima as raízes da equação na tela, os valores a, b e c são inseridos pelo usuário: ax2 + bx + c = 0, Δ= b2 - 4ac ,

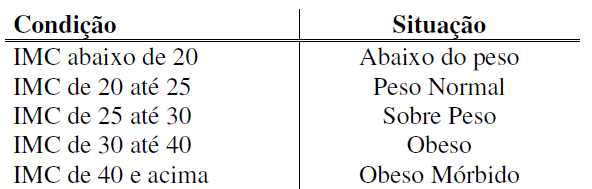
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Método | Rota | Envio de dados | Funcionalidade | Retorno |
| GET | /bhaskaras/a/b/c | /bhaskaras/3/15/12 | Retorna as raízes da equação | {  “a”:3,  “b”:15,  “c”:12,  “delta”:81,  “x1”:1,  “x2”:4  } |
| GET | /bhaskaras/ | {  “a”:3,  “b”:15,  “c”:12  } | Retorna as raízes da equação | {  “a”:3,  “b”:15,  “c”:12,  “delta”:81,  “x1”:1,  “x2”:4  } |

**\*obs.: para os exercícios 12 e 13 monte você a tabela de rotas, entrada de dados e retornos**

1. Construa programa para determinar se o indivíduo esta com um peso favorável. Essa situação é determinada através do IMC (Índice de Massa Corpórea), que é definida como sendo a relação entre o peso (PESO) e o quadrado da Altura (ALTURA) do indivíduo. Ou seja



Escreva na tela de acordo com as seguintes situações:



|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Método | Rota | Envio de dados | Funcionalidade | Retorno |
|  |  |  |  |  |

1. Construa programa que leia três lados de um triângulo, verifique e escreva que tipo de triângulo eles formam (considere **triângulo equilátero** com três lados iguais, **triângulo isósceles** com dois lados iguais e **triângulo escaleno** com todos os lados diferentes).

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Método | Rota | Envio de dados | Funcionalidade | Retorno |
|  |  |  |  |  |
|  |  |  |  |  |

1. Analise o modelo de banco de dados abaixo:

|  |
| --- |
|  |
| CREATE SCHEMA IF NOT EXISTS `PetPAW` DEFAULT CHARACTER SET utf8 ;  USE `PetPAW`;  CREATE TABLE IF NOT EXISTS `PetPAW`.`tipo` (  `idTipo` INT NOT NULL AUTO\_INCREMENT,  `nomeTipo` VARCHAR(45) NULL,  PRIMARY KEY (`idTipo`))  ENGINE = InnoDB;  CREATE TABLE IF NOT EXISTS `PetPAW`.`Raca` (  `idRaca` INT NOT NULL AUTO\_INCREMENT,  `nomeRaca` VARCHAR(45) NULL,  `idTipo` INT NOT NULL,  PRIMARY KEY (`idRaca`, `idTipo`),  INDEX `fk\_raca\_tipo1\_idx` (`idTipo` ASC),  CONSTRAINT `fk\_raca\_tipo1`  FOREIGN KEY (`idTipo`)  REFERENCES `PetPAW`.`tipo` (`idTipo`)  ON DELETE NO ACTION  ON UPDATE NO ACTION)  ENGINE = InnoDB;  CREATE TABLE IF NOT EXISTS `PetPAW`.`pet` (  `idpet` INT NOT NULL AUTO\_INCREMENT,  `nome` VARCHAR(45) NULL,  `idade` TINYINT NULL,  `obs` VARCHAR(45) NULL,  `idRaca` INT NOT NULL  PRIMARY KEY (`idpet`, `idRaca`),  INDEX `fk\_pet\_raca\_idx` (`idRaca` ASC),  CONSTRAINT `fk\_pet\_raca`  FOREIGN KEY (`idRaca`)  REFERENCES `PetPAW`.`Raca` (`idRaca`)  ON DELETE NO ACTION  ON UPDATE NO ACTION)  ENGINE = InnoDB; |

Construa uma api do tipo rest com as seguintes rotas:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Método | Rota | Envio de dados | Funcionalidade | Retorno |
|  |  |  |  |  |
| POST | petshops/tipos/ | {  “nomeTipo”:”Cachorro”  } | Cadastrar um tipo de animal | {  “cod”:”OK”,  “msg”:”Cadastrado com sucesso”,  “Cadastrado”: {  “idTipo”: 1,  “nomeTipo”:”Cachorro”  }  } |
| GET | petshops/tipos/ | “” | Retorna todos os tipos em ordem alfabética | [{  “idTipo”: 1,  “nomeTipo”:”Cachorro”  },  “idTipo”: 2,  “nomeTipo”:”gato”  }  ] |
| GET | petshops/tipos/id | petshops/tipos/1 | Retorna o tipo de animal com o id correspondente. | {  “idTipo”: 1,  “nomeTipo”:”Cachorro”  } |
| PUT | petshops/tipos/ | {  “idTipo”: 1,  “nomeTipo”:”Cachorro”  } | Altera o tipo | {  “cod”:”OK”,  “msg”:”alterado com sucesso”,  “alterado”: {  “idTipo”: 1,  “nomeTipo”:”Cachorro”  }  } |
| DELETE | petshops/tipos/id | petshops/tipos/1 | Exclui um tipo | {  “cod”:”OK”,  “msg”:”Excluído com sucesso”,  “excluido”: {  “idTipo”: “1”,  “nomeTipo”:””  }  } |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Método | Rota | Envio de dados | Funcionalidade | Retorno |
|  |  |  |  |  |
| POST | petshops/racas/ | {  “nomeRaca”:”vira-lata”,  “idTipo”:1  } | Cadastrar uma raça de animal | {  “cod”:”OK”,  “msg”:”Cadastrado com sucesso”,  “Cadastrado”: {  “idRaca”:1,  “nomeRaca”:”vira-lata”,  “idTipo”:1  }  } |
| GET | petshops/racas/ | “” | Retorna todas as raças em ordem alfabética | [{  “idRaca”:2,  “nomeRaca”:”Labrador”,  “idTipo”:1  “nomeTipo”:”Cachorro”  },  {  “idRaca”:1,  “nomeRaca”:”vira-lata”,  “idTipo”:1  “nomeTipo”:”Cachorro”  }] |
| GET | petshops/racas/id | petshops/racas/2 | Retorna uma raça com o id correspondente. | {  “idRaca”:2,  “nomeRaca”:”Labrador”,  “idTipo”:1  “nomeTipo”:”Cachorro”  } |
| PUT | petshops/racas/ | {  “idRaca”:2,  “nomeRaca”:”Labrador”,  “idTipo”:1  } | Altera uma raça | {  “cod”:”OK”,  “msg”:”alterado com sucesso”,  “alterado”: {  “idRaca”:2,  “nomeRaca”:”Labrador”,  “idTipo”:1  }  } |
| DELETE | petshops/racas/id | petshops/racas/2 | Exclui uma raça | {  “cod”:”OK”,  “msg”:”Excluído com sucesso”,  “excluido”:{  “idRaca”:2,  “nomeRaca”: null,  “idTipo”:null  “nomeTipo”:null  }  } |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Método | Rota | Envio de dados | Funcionalidade | Retorno |
| POST | /pet | {  “nome”:”Bob”,  “idade”: 3,  “obs”: “Bravo”,  “idRaca”:2  } | Cadastra um novo pet | {  “cod”:”OK”,  “msg”:”Cadastrado com sucesso”,  “Cadastrado”: {  “idPet”:3,  “nome”:”Bob”,  “idade”: 3,  “obs”: “Bravo”,  “idRaca”:2  }  } |
| GET | /pet | “” | Retorna todos os pets em ordem alfabética | [  {  “idPet”:3,  “nome”:”Bob”,  “idade”: 3,  “obs”: “Bravo”,  “idRaca”:2,  “nomeRaca”: “Labrador”  }  ] |
| GET | /pet/id | /pet/3 | Retorna o pet com id igual a 1 | {  “idPet”:3,  “nome”:”Bob”,  “idade”: 3,  “obs”: “Bravo”,  “idRaca”:2,,  “nomeRaca”: “Labrador”  } |
| PUT | /pet/ | {  “idPet”:3  “nome”:”Bob”  “idade”: 3  “obs”: “Bravo”  “idRaca”:2,  “nomeRaca”: “Labrador”  } | Atualiza o pet | {  “cod”:”OK”,  “msg”:”alterado com sucesso”,  “alterado”: {  “idPet”:3,  “nome”:”Bob”,  “idade”: 3,  “obs”: “Bravo”,  “idRaca”:2,  “nomeRaca”: “Labrador”  } } |
| DELETE | /pet/id | /pet/3 | Exclui o pet com id específico. | {  “cod”:”OK”,  “msg”:”excluído com sucesso”,  “excluido”: {  “idPet”:3,  “nome”:”Bob”,  “idade”: 3,  “obs”: “Bravo”,  “idRaca”:2,  “nomeRaca”: “Labrador”  } } |