Lista de Exercícios 2

1. Crie uma função que receba dois parâmetros: um array de strings e uma string. Remova todos elementos do array que são iguais a string passada como parâmetro (utilize o unset). Retorne o array como resultado.
2. Crie uma função que receba um array de strings e remova todos os elementos repetidos.
3. Crie uma função que receba como parâmetro: um array de inteiros ordenados crescentemente e um número inteiro. Crie um novo array com o novo inteiro inserido no local correto. Retorne esse array.
4. Crie uma função printTable que receba um array composto, sendo que cada elemento pode conter um número variável de colunas. Pegue o maior número de colunas e imprima uma tabela conforme o exemplo abaixo. No último elemento das linhas com número menores de colunas, utilize "colspan" para adequar o a tabela:

```
54 $a = array (
55       1 =>  array( "Joao da Silva", "Rondonopolis", 25 ),
56       2 =>  array( "Maria da Silva", "Rondonopolis", 25, "Brasil", "Futebol" ),
57       3 =>  array( "Jose da Silva", "Ciudad del Este", 23, "Paraguai" ),
58       4 =>  array( "Olavo da Silva", 26 )
59    );
60
61 printTable($a);
62 ?>
```

Resultado esperado:

Joao da Silva	Rondonopolis	25		
Maria da Silva	Rondonopolis	25	Brasil	Futebol
Jose da Silva	Ciudad del Este	23	Paraguai	
Olavo da Silva	26			

(utilize:

```
<style>
table, th, td {
    border: 1px solid black;
    border-collapse: collapse;
}
</style>
```

e

```
style="background: white" ou style="background: gray" para as linhas)
```

)

5. Crie uma função que faça um "merge" entre dois arrays ordenados e retorne o array resultante.
6. Crie um php que identifica se parâmetros de um formulário qualquer lhe foram enviados via POST ou GET (pode-se utilizar a função count).
7. Crie um formulário com os campos "nome", "idade" e "CPF". Direcione esse conteúdo para um script PHP que deve armazená-lo na sessão. Utilize o CPF para indexar o array utilizado para armazenar o conteúdo. Mostre todos os conteúdos que foram postados anteriormente.

8. Repita o exercício 7 criando uma classe “Pessoa” para armazenar o “nome”, “idade” e “CPF” (crie funções set/get para cada atributo privado). A cada novo “POST”, armazene os objetos na sessão e imprima todos já adicionados.
9. Crie uma função que receba um array de objetos “Pessoa” (vide exercício anterior) e retorne um novo array com os objetos ordenados por idade.
10. Crie as seguintes classes, atributos e respectivos métodos set/get:

	Classes			
	Pessoa	**Genitor**	**Filho**	**Familia**
Herda	–	Pessoa	Pessoa	–
Atributos	- Nome - Idade - CPF	- Filhos[]	- Pai : Genitor - Mãe : Genitor	- Pai : Genitor - Mãe : Genitor - Filhos[] : Filho
Métodos	+ set's/get's	+ adicionaFilho(...) + removeFilho(...) + getFilhos() + setFilhos(...)	+ set's / get's	+ set's / get's

11. Utilize as classes criadas no exercício 10 para criar um array de objetos Família's, sendo que cada família tem um Pai, Mãe e zero ou mais filhos.
12. Crie uma função que receba um array de famílias, imprimindo todos os dados de cada família: nome do pai, nome da mae, nome e idade de cada filho.
13. Implemente uma função que percorra o array de famílias e imprima todas as pessoas que possuem uma determinada idade.
14. Crie um jogo da velha utilizando 9 botões em uma tabela 3x3. Na primeira vez que o usuário acessa sua página, seu script PHP inicia com um X (utilize o label do próprio botão) aleatoriamente (utilize a função rand(min, max) do PHP para emular a jogada do computador). Cada vez que o usuário clicar em um botão que não tenha X e nem O, marque com um X a jogada do usuário e adicione um O aleatoriamente. Adicionalmente, implemente um método verificarGanhador() que deve ser invocado a cada jogada para verificar se existe algum ganhador. Utilize orientação a objeto e sessão para manter as jogadas. Quando houver um ganhador, avise o usuário e destrua a sessão.
15. Considere os seguintes estilos e HTML:

Estilo	HTML

.retangulo {
width: 200px;
height: 100px;
background: black;
}
 	
.circulo {
width: 100px;
height: 100px;
background: red;
-moz-border-radius: 50px;
-webkit-border-radius: 50px;
border-radius: 50px;
}
 	
.quadrado {
width: 100px;
height: 100px;
background: blue;
}
 	
.triangulo {
width: 0;
height: 0;
border-left: 50px solid transparent;
border-right: 50px solid transparent;
border-bottom: 100px solid yellow;
}
	Label
	
	 Label
	
	 Label
	
	 Label

Cada uma das “classes de estilo” corresponde a uma figura geométrica. Crie uma classe base abstrata chamada FiguraGeometrica que possui o seguinte método abstrato: desenhar( ) e o seguinte atributo privado Label. Crie as classes Retangulo, Circulo, Quadrado e Triangulo. Cada uma deve implementar o método desenhar, que retorna o HTML que desenhe a figura geométrica

correspondente e imprime o Label dentro da Figura. Teste suas classes instanciando cada um e desenhando na tela as figuras.

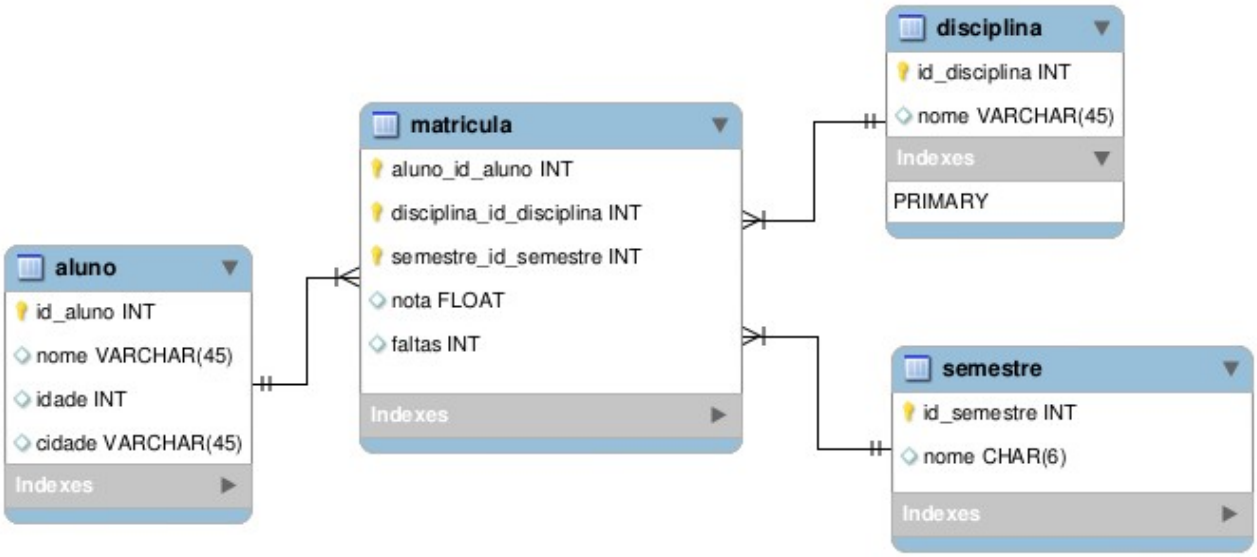
16. Utilizando as classes do exercício anterior e crie uma página que o usuário escolhe uma figura geométrica por meio de um “ComboBox” e adicione essa figura a um array armazenado em sessão. Desenhe todos os objetos já armazenados na sessão.
17. Utilizando os métodos mágicos _set e _get, crie uma classe chamada “Registro” que armazene quaisquer chaves e seus respectivos valores. Implemente também um método que imprima todas as chaves armazenadas e seus valores.

Script Exemplo	Saída Esperada

registro = new Registro();
registro->carro = “Maverick”;
registro->xyz = “XYZ”;
registro->numero = 1;
echo registro->carro . “
”;
echo registro->xyz . “
”;
registro->imprimirTodos();
	Maverick
	XYZ
	• carro: Maverick
	• xyz: XYZ
	• numero: 1

18. Utilize o SQL passado no AVA para criar um DataBase no MySql. O modelo está a seguir. A partir dele desenvolva as Classes para cada tabela e Classes DAO para acesso a base de dados, cada uma contendo métodos inserir, remover, selecionar e salvar. Além disso, implemente:

(a) Página para: Matrícula de alunos em disciplinas, para um determinado semestre (pré-selecionado)
(b) Página para: Lançamento de notas e faltas em disciplinas, para um determinado semestre (pré-selecionado)
(c) O layout de sua página inicial deve ter como menu uma lista de todos os semestres disponíveis. A partir do clique no semestre, o usuário poderá escolher se deseja efetuar uma matrícula ou lançamento de notas.
(d) Implemente uma página que liste todas as disciplinas. Cada disciplina deve ser um link para uma segunda página. Essa segunda página deve listar todos os nomes dos alunos matriculados naquela disciplina e os nomes do semestre correspondente.
(e) Faça uma página para cadastro de alunos
(f) Faça uma página para cadastro de semestre
(g) Faça uma página para cadastro de disciplinas
(h) Faça uma página que liste todos os alunos. Cada nome de aluno é um link que abre uma segunda página. Essa segunda página deve conter todas as disciplinas e o semestres que esse aluno está/esteve matriculado.
(i) Faça uma página para Listar toda disciplina/semestre. Cada um desses itens deve ser um link para abrir uma segunda página para selecionar um aluno. Ao selecionar esse aluno, deve-se abrir uma terceira página para atribuir nota e falta.
(j) Faça uma listagem de todos os alunos e suas notas e faltas.


disciplina
id_disciplina INT
nome VARCHAR(45)
Indexes
PRIMARY
matricula
aluno_id_aluno INT
disciplina_id_disciplina INT
semestre_id_semestre INT
nota FLOAT
faltas INT
Indexes
aluno
id_aluno INT
nome VARCHAR(45)
idade INT
cidade VARCHAR(45)
Indexes
semestre
id_semestre INT
nome CHAR(6)
Indexes