

Nome _____		Registro Acadêmico _____
Curso _____	Semestre - Turma – Bimestre _____	Data _____
Professor(a) _____	Disciplina _____	Visto _____

INSTRUÇÕES GERAIS

- Avaliação individual e sem consulta a livros ou anotações. Mantenha postura adequada.
- Desligue seu aparelho celular. Não será permitido seu uso durante a avaliação.
- Utilize caneta esferográfica azul ou preta. Questões a lápis não serão objeto de revisão.
- Preencha seu Nome e R. A. no cabeçalho das folhas de questões e de respostas;
- Não será aceita a entrega da prova antes de **meia hora** de seu início.

Lista de Exercícios

- **Esta lista de exercícios é individual (não é em grupos).**
- **A lista deverá ser entregue até o dia 19/06/2007.**

1) Dado um arquivo de entrada (peças.txt) contendo o código de uma peça no formato AA-9999 e a quantidade pedida, gere um arquivo chamado lista_de_peças.txt com o consolidado das peças.

Arquivo de entrada: peças.txt

Formato do arquivo de entrada:

```
AA-0004,13
AA-0002,30
AA-0001,20
AA-0001,15
AA-0004,15
```

A saída do programa deverá exibir os códigos em ordem alfabética e consolidado.

Peça	Qtd
AA-0001	35
AA-0002	30
AA-0004	28

2) Escreva uma função que calcule as raízes de uma equação de 2º grau usando a fórmula de Baskara:

Uma equação de 2º Grau possui o seguinte formato:

$$A*x^2 + b*x + c = 0$$

Onde a, b e c são chamados coeficientes da equação.

```
delta=(b*b-4*a*c)
```

As raízes da equação (quando houverem) são calculadas em x1 e x2.

```
x1 = -b - sqrt(delta)
-----
      2*a
```

```
x2 = -b + sqrt(delta)
-----
      2*a
```

Se delta for maior que zero então a equação terá duas raízes.

Se delta for igual a zero então a equação terá uma única raiz.

Se delta for negativo então a equação não possui raízes.^a

Por exemplo, com a=1, b=-5 e c=6 as raízes são 2 e 3.

A função deverá retornar as duas raízes caso elas existam. Caso apenas um raiz exista, essa deverá retornar em x1 e x2 deverá ter o valor undef.

Caso a equação não possua raízes então a função deverá retornar undef tanto para x1 quanto para x2.

```
($x1,$x2) = calcula_raizes(1,-5,6)
```

```
1,0,-25
```

```
1,1,1
```

```
1,0,0
```

Escreve um script que solicite os três coeficientes: a, b e c e utilize a função calcula_raizes para calcular as raízes da equação.

Exemplo:

```
Entre com a: 1
```

```
Entre com b: -5
```

```
Entre com c: 6
```

```
A equação possui duas raízes:
```

```
X1=2
```

```
X2=3
```

```
Entre com a: 1
```

```
Entre com b: 0
```

```
Entre com c: 0
```

```
A equação possui uma única raiz:
```

```
X1=0
```

```
X2=undef
```

3) Escreva um script em Perl que leia uma lista de nomes fornecidas pelo usuário (teclado) e imprima uma lista contendo os nomes em ordem decrescente. Assim como a função sort há a função reverse que cria uma lista em ordem reversa.

4) Escreva um programa um programa que seja uma pequena calculadora. Inicialmente deverá ser criado uma estrutura de pilha que poderá ser implementado usando as funções push (insere um elemento no inicio da lista) e pop (retorna o elemento no topo da lista). A cada número digitado insira o

^a A equação não terá raízes com conjunto de números reais.

número na pilha (início da pilha). Quando você digitar um operador (+,-,*,/) então será necessário remover os dois elementos do topo da pilha (remover dois elementos do início na lista) e usar o operador fornecido para calcular o resultado que será novamente inserido no topo da pilha (início da lista).

Exemplo de execução:

```
-> 1
-> 2
-> +
Topo: 3
-> 4
-> 5
-> *
Topo: 20
-> -
Topo: 17
->
```

5) Escreva um script que solicite a temperatura e que imprima a mensagem segundo a temperatura fornecida

```
Temperatura <= 0 : Está muito frio ...
Temperatura > 0 e Temperatura < 10 : Temperatura fria mas ainda pode piorar
Temperatura >=10 e Temperatura < 20 : Temperatura agradável
Temperatura >= 20 e Temperatura < 30 : Temperatura está esquentando
Temperatura >= 30: Está muito quente ... Haja ar condicionado.
```

Exemplo de execução:

```
Entre com a temperatura: -4
> Está muito frio (-4°C)
Entre com a temperatura: 0
> Temperatura fria mas ainda pode piorar (0°C)
Entre com a temperatura: 19
> Temperatura agradável (19°C)
Entre com a temperatura: 25
> Temperatura está esquentando ... (25°C)
Entre com a temperatura: 30
> Está muito quente ... Haja ar condicionado (30°C)
```

6) Escreva um programa que gere uma tabela de temperaturas em graus Celsius e graus Fahrenheit como indicado abaixo:

```
0°C - 32.00°F
1°C - 33.80°F
2°C - 35.60°F
3°C - 37.40°F
...
100°C - 212.00°F
```

A fórmula de conversão é: $TF = (9/5) * TC + 32$ onde:

TF = Temperatura em °F

TC = Temperatura em °C

7) Escreva um script que leia o arquivo /etc/passwd e escreva-o em um arquivo chamado passwd.csv. Um arquivo .csv possui todos seus campos separados por ','. Lembre-se que o arquivo /etc/passwd possui seus campos separados por ':'

Dica: A função `join` pega expressão (EXPR) e uma lista de valores e retorna uma string inserindo EXPR entre cada elemento da lista.

Exemplo:

```
@lista=(a,b,c,d);
$novaString=join(":",@lista);

$novaString="a:b:c:d";
```

8) A função `localtime` retorna uma lista de nove elementos com informações sobre a data e hora atual. É tipicamente usado como:

```
($s,$min,$hora,$mdia,$mes,$ano,$wday,$yday,$ehDst)=localtime;
```

Onde:

- `$s`: número de segundos
- `$min`: número de minutos
- `$hora`: número de horas
- `$mdia`: número do dia
- `$mes`: retorna um número de 0 a 11 indicando o mês. Sendo 0="janeiro" e 11="dezembro"
- `$ano`: é o número de anos desse 1900. Portanto, para obter o ano atual acrescente 1900 ao valor retornado.
- `$wday`: retorna um número entre 0 e 6 sendo Domingo igual a 0, Segunda-feira igual a 1 e Sábado igual a 7.
- `$yday`:
- `$ehDst`:

a) Escreva uma função (`retorna_data`) que retorne uma string no seguinte formato:

Campinas, 08 de junho de 2007. São 10:01.

Exemplo:

```
print retorna_data(),"\n";
```

9) Escreva dois scripts, um cliente (`cli_hora.pl`) e servidor (`ser_hora.pl`). O servidor, quando solicitado pelo cliente, deverá retornar a data e hora local no formato: `dd/mm/aaaa hh:mi:ss`.

O cliente deverá solicitar ao servidor a hora atualizada e imprimir a seguinte mensagem:

Data atual é: `dd/mm/aaaa`

Hora atual é: `hh:mi:ss`

10) A linguagem Perl permite que façamos vários testes a respeito de arquivos, como: verificar a existência de um arquivo, verificar se um diretório pode ser lido ou escrito, etc.

Veja o exemplo:

```
$nome="dados.txt";
if(-e $nome) {
    print "você já possui um arquivo chamado $dados no diretório corrente\n";
}
```

Veja um subconjunto da lista de testes abaixo na tabela:

-r	Arquivo ou diretório possui permissão de leitura
-w	Arquivo ou diretório possui permissão de escrita
-x	Arquivo ou diretório é executável
-e	Arquivo ou diretório existe
-d	Entrada é um diretório
-M	Número de dias da última modificação do arquivo
-A	Número de dias do último acesso ao arquivo

Escreva um script que solicite que o usuário entre com um nome de uma entrada no diretório (arquivo ou diretório) e liste as seguintes características:

- se o arquivo ou diretório possui permissão de leitura
- se o arquivo ou diretório possui permissão de escrita
- se a entrada é um arquivo ou diretório
- se o arquivo foi acessado ou modificado nos últimos 7 dias.

Exemplo:

```
maurocl@d520:~/policamp/prova02$ ./questao10.pl
Entre com o nome do arquivo: /home/maurocl
diretorio /home/maurocl possui permissao de leitura
diretorio /home/maurocl possui permissao de escrita
diretorio /home/maurocl NAO foi modificado ha mais de uma semana
diretorio /home/maurocl NAO foi acessado a ha mais de uma semana

maurocl@d520:~/policamp/prova02$ ./questao10.pl
Entre com o nome do arquivo: questao10.pl
arquivo questao10.pl possui permissao de leitura
arquivo questao10.pl possui permissao de escrita
arquivo questao10.pl NAO foi modificado ha mais de uma semana
arquivo questao10.pl NAO foi acessado a ha mais de uma semana
```

11) Escreva um programa que solicite ao usuário um arquivo de entrada e um arquivo de saída. O programa deverá copiar o arquivo de entrada no arquivo de saída. No final os dois arquivos deverão ter conteúdos idênticos.

12) Escreva um programa que leia o nome de um arquivo do STDIN e então abra esse arquivo e exiba seu conteúdo numerando as linhas 001, 002, ... Por exemplo, se o arquivo saída.txt contiver as seguintes linha:

```
arquivo: x.txt
Nome
Endereço
Cidade - SP
CEP
```

Ele irá exibir:

```
001: Nome
002: Endereço
003: Cidade - SP
004: CEP
```

13) Escreva uma subrotina (chamada ConvNumNome) que receba como argumento um valor entre 0 e 9 e retorne o número em Português (como: 0 => "zero", 1 => "um", 2 => "dois", 3 => ...)

Agora, escreva um programa que solicite um número de até 4 dígitos e use a subrotina anteriormente escrita e para exibir cada número escrito por extenso.

Exemplo:

```
$ ./extenso.pl
Entre com o número: 1234
Um dois três quatro
Entre com o número: 1945
Um nove quatro cinco
Entre com o número: 1
Um
Entre com o número: 987
Nove oito sete
```

Dica: use a função `length($string)` para obter o tamanho da string.

14) Escreva um programa que leia uma lista de números (em linhas separadas) até que o número 999 seja lido e imprima o total da soma de todos os números. Tenha certeza que o número 999 não seja adicionado à lista. Por exemplo, se você entrar com os números: 1, 2, 3 e 999 o programa deve imprimir a resposta 6 (1+2+3).

15)Veja o script abaixo:

```
#!/usr/bin/perl
#
# questao15.pl
#
$a=3;
@a=(1,2,3);
%a=( 1 => "um", 2 => "dois", 3 => "três");

print $a[$a];
print @a;
print $a{$a};
```

Execute o script:

```
$/questao15.pl
```

Agora, explique:

Qual é a diferença entre `$a`, `@a` e `%a` ? O que representa os símbolos `$`, `@` e `%` prefixados ao nome de uma variável ?

16) O Perl possui duas funções cujos nomes são bem parecidos: `chop` e `chomp`. Veja o programa abaixo:

```
print "entre com um nome: "
$nome=<STDIN>;
print chop($nome);
print chomp($nome);

print chop($nome);
print chomp($nome);
```

Explique a diferença entre o uso de uso das duas funções.

17) Veja a saída do programa abaixo:

```
while(<STDIN>) {  
    print "[$_]\n";  
}
```

Corrija o programa acima de forma que ele não pule uma linha antes do caractere `]`

18) Veja o programa abaixo:

```
$a = 1;  
$b= $a++  
$a = 1;  
$c= ++$a;  
  
print "\$a=$a\n";  
print "\$b=$b\n";  
print "\$c=$c\n";
```

- a) Quais os valores das variáveis escalares a, b e c ?
- b) Explique as diferenças de valores entre as variáveis \$b e \$c