Lista de Exercícios 09

Estruturas de Repetição

Exercício 01) Escreva programas em *shell script* para cada um dos itens a seguir. **Cada *script* deve usar obrigatoriamente o comando** `for`.

a) Mostre os sete dias de semana usando código de três dígitos indicando os que são dia de semana e quais são fim de semana.

```
$ ./days
Mon (weekday)
Tue (weekday)
Wed (weekday)
Thu (weekday)
Fri (weekday)
Sat (weekend)
Sun (weekend)
```

```
#!/bin/bash

for day in "Mon (weekday)" "Tue (weekday)" "Wed (weekday)" "Thu (weekday)" "Fri (weekday)" "Sat (weekend)" "Sun (weekend)"; do
    echo "$day"
done
```

b) Mostre todos os usuários do arquivo `/etc/passwd`.

```
$ ./users
User 1: root
User 2: daemon
User 3: nobody
...
```

```
#!/bin/bash

for (( i=1; i <= $(cat /etc/passwd | wc -l); i++ )); do
    echo "User $i: $(cat /etc/passwd | cut -f 1 -d":" | sed -n "$i p")"
done
```

c) Sorteie `n` rodadas de dados de `1` a `6`, definidas pelo usuário via argumentos em linha de comando, e imprima a frequência de todos os lados.

```
$ ./dicesfreq 25
Dice 1: 4
Dice 2: 5
Dice 3: 7
Dice 4: 2
Dice 5: 4
Dice 6: 3
```

```
#!/bin/bash
um=0
dois=0
tres=0
quatro=0
cinco=0
seis=0
for (( i=0; i < "$1"; i++ )); do
	dice="$(( $RANDOM % 6 + 1))"
	case "$dice" in
		1)
			$((um++))
			;;
		2)
			$((dois++))
			;;
		3)
			$((tres++))
			;;
		4)
			$((quatro++))
			;;
		5)
			$((cinco++))
			;;
		6)
			$((seis++))
			;;
	esac
done
echo "Dice 1: $um"
echo "Dice 2: $dois"
echo "Dice 3: $tres"
echo "Dice 4: $quatro"
echo "Dice 5: $cinco"
echo "Dice 6: $seis"
```

d) Receba um ou mais números via argumentos em linha de comando e mostrar o menor e o maior deles. Restrição: não é permitido usar o comando `sort` nem outro comando similar.

```
$ ./edges 10 7 81 40 74 22 15 21 84 74
Lower: 7
Higher: 84
```

```
#!/bin/bash

maior="$1"
menor="$1"

for i in $@; do
        if (( maior < "$i" )); then
                maior="$i"
        elif (( menor > "$i")); then
                menor="$i"
        fi
done

echo "Lower: $menor"
echo "Higher: $maior"
```

Exercício 02) Escreva um *shell script* para treinar um usuário com operações básicas de aritmética: soma e subtração. Esse *script* poderia ser colocado na inicialização de cada terminal, obrigando o usuário a informar o resultado correto para ter acesso a ele. O *script* deve gerar dois números aleatórios entre `1` e `999` (inclusivos) que serão usados como operandos. O *script* também deve gerar aleatoriamente um dos operandos de adição (`+`) ou subtração (`-`). O *script* deve imprimir exatamente conforme exemplo a seguir, alinhando os números à direita. Usuários não podem abortar o programa com `CONTROL+C`, `CONTROL+D` ou `CONTROL+Z`. **O *script* deve usar obrigatoriamente o comando** `while`. Dica: use o comando `printf` para imprimir a conta e o comando `expr` para calcular o resultado.

```
$ ./basic_math
     43
  - 287
  -----
? -143
Wrong answer
? -244
Right answer
$
```

```
#!/bin/bash

operadorEscolha="$RANDOM"
operando1=$(( ( RANDOM % 999 ) + 1 ))
operando2=$(( ( RANDOM % 999 ) + 1 ))
resposta=0

if (( "$operadorEscolha" % 2 == 0 )); then
    operador="-"
else
    operador="+"
fi

resultado=$(( "$operando1" "$operador" "$operando2" ))

echo "$resultado"

printf "%5d\n%s% 3d\n-----\n" "$operando1" "$operador" "$operando2"
```

```
trap 'echo " Hj não malandro"' SIGINT SIGTSTP

while true; do
        read -p "? " resposta
        if ((resposta == resultado)); then
                echo "Right answer"
                break
        else
                echo "Wrong answer"
        fi
done
```

Exercício 03) Escreva um jogo de adivinhação de um número em *shell script*. O programa deverá selecionar um número aleatório entre `1` e `1000` (inclusivos) sem informar ao usuário qual. Se o usuário acertar o número, o programa deve informar que o usuário venceu; caso contrário, o programa deve informar se o número sorteado é menor ou maior que aquele escolhido pelo usuário. O jogo continua indefinidamente até que o usuário acerte o número. **O *script* deve usar obrigatoriamente o comando** `until`. Dica: não é preciso fazer validação se o valor digitado é numérico.

```
$ ./guess
Try to guess the number I thought!

Choose a number (1 to 1000): 500
  My number is higher than 500
Choose a number (1 to 1000): 750
  My number is lower than 750
Choose a number (1 to 1000): 625

Congratulations, you guessed correctly
$
```

```
#!/bin/bash

escolhido=$(( ("$RANDOM" % 1000) + 1))

echo "Try to guess the number I thought!"
echo

until false; do
    read -p "Choose a number (1 to 1000): " resposta

    if (( "$resposta" == "$escolhido" )); then
        echo
        echo "Congratulations, you guessed correctly"
        break
    elif (( "$resposta" < "$escolhido" )); then
        echo "My number is higher than $resposta"
    else
        echo "My number is lower than $resposta"
    fi
done
```

Exercício 04) Desenvolva um *shell script* para verificar se uma frase é um palíndromo, imprimindo se sim ou não. O programa deve receber da entrada padrão frases em cada linha e deve terminar quando receber uma frase vazia. Você deve considerar apenas os caracteres do alfabeto inglês (sem acentos), ignorando todos os outros. Dica: use o comando `sed` para filtrar os caracteres inválidos `sed 's/[^A-Za-z]//g'`.

```
$ cat input.txt
reviver
lamina animal
Ser ou nao ser?
A Daniela ama a lei? Nada!
$ ./palindrome < input.txt
Yes
Yes
No
Yes
```

```
#!/bin/bash

while read -r linha; do
    if [[ -z "$linha" ]]; then
        break
    fi

    processada=$(echo "$linha" | sed 's/[^A-Za-z]//g' | tr '[:upper:]' '[:lower:]')

    invertida=$(echo "$processada" | rev)

    if [[ "$processada" == "$invertida" ]]; then
        echo "Yes"
    else
        echo "No"
    fi
done
```