ATIVIDADE 3 - PROBLEMAS PROPOSTOS

Utilizando o Portugol Studio ou o Portugol Webstudio, codifique os algoritmos para as situações abaixo. Leia e releia com atenção os enunciados. Os exemplos são mais diretos que o desejável em implementação – ou seja, fique livre para incluir mensagens de orientação ao usuário durante a execução do algoritmo.

1) No restaurante do hotel é oferecida uma cortesia para os hospedes. Em toda refeição o hotel paga R$ 30,99 reais do valor gasto pelos hospedes. Caso o hospede gaste menos que R$ 30,99 ou exatamente esse valor ele não precisa pagar nada. Caso o valor da mesa seja maior que R$ 30,99 o hóspede precisa pagar a diferença (o valor da refeição descontando a cortesia).
No sistema primeiro precisamos solicitar para o usuário e armazenar o valor de 4 mesas e guardar dentro de um vetor esses valores. Posteriormente precisamos verificar os valores armazenados para retornar para o usuário se a mesa precisa pagar ou não algum valor. Caso nenhum valor precise ser pago deve ser retornada a mensagem "Nada a pagar!", caso a mesa precise pagar algum valor deve ser retornado "A mesa precisa pagar: " e logo em seguida ser retornado quanto a mesa deve. Lembrando que precisamos fazer isso para todas as mesas.

Exemplo:

```
[O seu sistema solicitou]
Valor da mesa 1:
[O usuário respondeu]
30
[O seu sistema solicitou]
Valor da mesa 2:
[O usuário respondeu]
35
[O seu sistema solicitou]
Valor da mesa 3:
[O usuário respondeu]
45
```

```
[O seu sistema solicitou]
Valor da mesa 4:
[O usuário respondeu]
25
[O seu sistema respondeu]
Mesa 2 precisa pagar R$ 4,01
Mesa 3 precisa pagar R$ 14,01
```

2) **Deve ser utilizado apenas um vetor de 10 posições.** Considerando que o hotel tenha 10 quartos, desenvolva um algoritmo para marcar a ocupação de cada quarto. Todos os quartos iniciam como livres, o usuário informará então o número do quarto (de 1 a 10). Internamente o número do quarto precisa ser de 0 até 9, então altere o valor informado pelo usuário para respeitar isso. O sistema questionará “O quarto está livre ou ocupado? (L/O)”; o usuário informará L ou O e o sistema registrará essa escolha para o quarto. Existem as seguintes regas:

- Se o quarto está livre e o usuário digita L deve ser exibida a mensagem “quarto já está vazio”.
- Se o quarto está livre e o usuário digita O deve ser exibida a mensagem “quarto foi ocupado” e o valor do vetor deve ser alterado.
- Se o quarto estiver ocupado e o usuário digitar L deve ser exibida a mensagem “quarto foi liberado” e o valor do vetor deve ser alterado.
- Se o quarto estiver ocupado e o usuário digitar O deve ser exibida a mensagem “quarto já está ocupado”.

Pergunte ao usuário se ele deseja continuar e caso ‘S’, repita a operação. Quando o usuário informar ‘N’ a repetição encerra e deve ser exibido o número do quarto (1 a 10) e a ocupação deste (O ou L). Exemplo:

```
[O usuário digitou]
5 (número do quarto)
O (livre ou ocupado)
[O seu sistema respondeu]
```

Deseja continuar? (S/N)

[O usuário digitou]

S (continuar ou não)

7 (número do quarto)

O (livre ou ocupado)

[O seu sistema respondeu]

Deseja continuar? (S/N)

[O usuário digitou]

S (continuar ou não)

5 (número do quarto)

O (livre ou ocupado)

[O seu sistema respondeu]

Quarto já ocupado

Deseja continuar? (S/N)

[O usuário digitou]

S (continuar ou não)

20 (número do quarto)

O (livre ou ocupado)

[O seu sistema respondeu]

Deseja continuar? (S/N)

S (continuar ou não)

20 (número do quarto)

L (livre ou ocupado)

[O seu sistema respondeu]

Quarto foi liberado

Deseja continuar? (S/N)

[O usuário digitou]

N (continuar ou não)

[O seu sistema respondeu]

```
 1 - L   2 - L   3 - L   4 - L   5 - O
 6 - L   7 - O   8 - L   9 - L  10 - L
11 - L  12 - L  13 - L  14 - L  15 - L
16 - L  17 - L  18 - L  19 - L  20 - L
```

3) Monte um algoritmo em que o usuário poderá cadastrar e pesquisar hóspedes. O algoritmo deve oferecer um menu com três opções ao usuário:
1- cadastrar; 2- pesquisar; 3- sair.
O algoritmo deve permitir que o usuário realize essas operações repetidas vezes, até que ele digite a opção “3”, que encerra o algoritmo. Sempre que o usuário decide repetir o código deve ser solicitado novamente o que ele deseja fazer. A pesquisa deve funcionar mesmo que poucos ou nenhum nome tenha sido cadastrado, realize a pesquisa até última posição preenchida.
A opção “cadastrar” deve permitir que o usuário informe um nome de hóspede, gravando-o em memória. O usuário só pode cadastrar até no máximo 7 hospedes, caso o número de hospedes seja maior que sete não deve ser solicitado o novo nome e uma mensagem (“Máximo de cadastros atingido”) deve ser exibida.
A opção “Pesquisar” deve permitir que ousuário informe um nome e, caso seja encontrado um nome exatamente igual, mostre a mensagem “Hospede (nome) foi encontrado no índice” (posição do vetor onde foi encontrado). **Todas as posições preenchidas do vetor devem ser verificadas**. Caso não tenha sido encontrado o nome em nenhuma das posições deve ser exibida a mensagem de “Hospede não encontrado”.

Exemplo:

```
[O seu sistema respondeu]
Digite 1- cadastrar; 2- pesquisar; 3- sair
[O usuário digitou]
1 (opção)
Maria da Silva (nome do hóspede)
[O seu sistema respondeu]
Digite 1- cadastrar; 2- pesquisar; 3- sair
[O usuário digitou]
1 (opção)
José Freitas (nome do hóspede)
[O seu sistema respondeu]
```

Digite 1- cadastrar; 2- pesquisar; 3- sair
[O usuário digitou]
2 (opção)
José Freitas (nome do hóspede)
[O seu sistema respondeu]
Hóspede José Freitas foi encontrado no índice 1
[O seu sistema respondeu]
Digite 1- cadastrar; 2- pesquisar; 3- sair
[O usuário digitou]
3