1)

A modularização é uma técnica de programação que permite a divisão da solução de um problema, afim de diminuir a complexidade, tornar o código mais organizado e evitar a repetição de códigos. Em cada declaração da função alguns parâmetros são obrigatórios e outros opcionais, veja cada parâmetro: tipo de retorno da função, o nome da função, parênteses após o nome da função, parâmetros, comandos e o tipo de retorno da função.

Neste contexto, complete as lacunas da sentença a seguir:

Quando o tipo de retorno da função for do tipo *void*  esse \_\_\_\_\_\_\_\_\_\_\_\_ não precisa ser usado, porém, quando não for *void*  é \_\_\_\_\_\_\_\_\_\_\_\_. O valor a ser retornado em uma função tem que ser \_\_\_\_\_\_\_\_\_\_\_\_ com o tipo de retorno, senão o problema dará um erro de \_\_\_\_\_\_\_\_\_\_\_\_ em algumas linguagens, em outras retornará um valor errôneo. Na linguagem C, deverá ser retornado um valor de acordo com o \_\_\_\_\_\_\_\_\_\_\_\_ de retorno da função.

Assinale a alternativa que completa as lacunas corretamente:

**Alternativas:**

* a)

vetor / permitido / provável / programação / valor

* b)

valor / válido / igualitário / estrutura / fator

* c)

programa / provável / condizente / acesso / comando

* d)

parâmetro / obrigatório / compatível / compilação / tipo

Alternativa assinalada

* e)

arquivo / assentido / favorável / otimização / aspecto

2)

De acordo com Mizrahi (2008) uma das definições de função é que ela é uma sub-rotina usada em um programa. Na linguagem de programação C, denominamos função a um conjunto de comandos que realiza uma tarefa específica em um módulo dependente de código. Uma função é referenciada pelo programa principal através do nome atribuído a ela. A utilização de funções visa modularizar um programa, o que é muito comum em programação estruturada. Desta forma podemos dividir um programa em várias partes, no qual cada função realiza uma tarefa bem definida.

A sintaxe de uma função:

*tipo\_de\_retorno nome\_da\_função (listagem de parâmetros)*

*{*

*instruções;*

*retorno\_da\_função;*

*}*

Uma função é definida como um trecho de código escrito para solucionar um subproblema. Esses blocos são escritos tanto para:

**Alternativas:**

* a)

aumentar a quantidade de linhas do programa, quanto para facilitar o trabalho do programador.

* b)

evitar erros de compilação, quanto para depurar o programa afim de achar os futuros erros de código e lógica.

* c)

dividir a complexidade de um problema maior, quanto para evitar a repetição de códigos.

Alternativa assinalada

* d)

possibilitar o uso de ponteiros em funções que possuem vetores, quanto funções que não possuem vetores.

* e)

usar funções com passagem de parâmetros por valor, quanto usar passagem de parâmetros por referência.

3)

As variáveis em C podem ser declaradas basicamente de três maneiras diferentes: dentro de uma função, fora de uma função, e como parâmetro de uma função. Essas três maneiras de declaração fazem com que as variáveis sejam chamadas de locais, globais ou parâmetros formais.

 Fonte:disponível em<http://www.di.ufpb.br/liliane/aulas/escopo.html>Acesso06.Ago.2018.

 Nesse contexto, julgue as seguintes afirmações

 I. As variáveis globais existem durante a execução de todo o programa.

 II. Uma variável local só pode ser utilizada pela função ou bloco que a declarou.

III. Os parâmetros formais são variáveis locais de uma função que são inicializadas no momento da chamada da função.

É correto apenas o que se afirma em:

**Alternativas:**

* a)

I.

* b)

III.

* c)

I e II.

* d)

II e III.

* e)

I, II e III.

Alternativa assinalada

4)

Embora a sintaxe da função recursiva seja similar as não recursivas, o funcionamento de ambas é bastante distinto e o mau uso dessa técnica pode acarretar em uso indevido de memória, muitas vezes chegando a travar a aplicação e o sistema. Para entender o processo,julgue as seguintes afirmações.

 I - A função recursiva chama a si própria até que um ponto de parada seja estabelecido, podendo ser alcançado através de uma estrutura condicional ou através de um valor informado pelo usuário.

 II -  No uso de uma função recursiva, os recursos são alocados em outro local da memória, ou seja, para cada chamada da função, novos espaços são destinados a execução do programa. E é justamente por esse ponto que o ponto de parada é crucial.

III - As variáveis criadas em cada instância da função na memória são dependentes, ou seja, quando as variáveis possuem nomes iguais, cada uma terá seu próprio endereço de memória, mas a alteração do valor em uma afetará a outra.

É correto apenas o que se afirma em:

**Alternativas:**

* a)

I.

* b)

II.

* c)

III.

* d)

I e II.

Alternativa assinalada

* e)

II e III.