socrative

Nome _____
Data _____

EDI - Lista Dupla 2/2

Nota _____

1.

```
1  void inserirFimUltimo(dadosDupla **ultimoRegistro){
2      //Criar o espaço de memória
3      dadosDupla *novo = (dadosDupla*) malloc (sizeof(dadosDupla));
4
5      //Preencher dados básicos
6      novo->codigo = RANDOMICO(1000);
7      inserirNome(novo->nome);
8
9      //Ajustar os ponteiros
10     novo->proximo = NULL; //Último registro, depois dele não tem ninguém.
11     novo->anterior = (*ultimoRegistro);
12     if ((*ultimoRegistro) != NULL) {
13         (*ultimoRegistro)->proximo = novo;
14     }
15     (*ultimoRegistro) = novo;
16 }
```

O código anterior efetua a inserção de um registro em uma lista duplamente encadeada, com inserção pelo fim, porém, Rogério, um desenvolvedor iniciante na linguagem C acredita que o if, escrito na linha 12, pode ser retirado, pois não tem utilidade no código, e, assim, o código funcionará normalmente.

(V) Verdadeiro

(F) Falso

2.

```
dadosDupla* funcao(dadosDupla* inicioLista, int cod) {
   dadosDupla* tmp;
   for (tmp = inicioLista; tmp != NULL; tmp = tmp→proximo) {
      if (tmp→codigo == cod) {
         return tmp;
      }
   }
  return NULL;
}
```

Considerando o código anterior é possível afirmar que:

(A) A função efetua a busca de um código a partir do início da lista, caso o registro seja encontrado será retornado, caso contrário, retornará NULL.

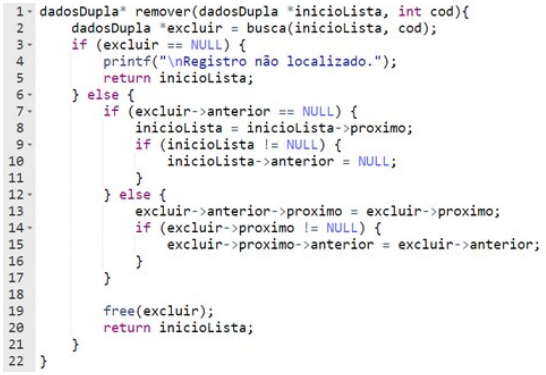
(B) A função efetua a inserção de um registro pelo fim de uma lista duplamente encadeada.

(C) A função efetua a inserção de um registro pelo início de uma lista simplesmente encadeada.

(D) A função efetua a remoção de um registro em uma lista duplamente encadeada.

(E) A função efetua a impressão de registros em uma lista simplesmente encadeada.

3. Sobre o código da figura é possível afirmar que:



```
1  dadosDupla* remover(dadosDupla *inicioLista, int cod){
2      dadosDupla *excluir = busca(inicioLista, cod);
3      if (excluir == NULL) {
4          printf("\nRegistro não localizado.");
5          return inicioLista;
6      } else {
7          if (excluir->anterior == NULL) {
8              inicioLista = inicioLista->proximo;
9              if (inicioLista != NULL) {
10                 inicioLista->anterior = NULL;
11             }
12         } else {
13             excluir->anterior->proximo = excluir->proximo;
14             if (excluir->proximo != NULL) {
15                 excluir->proximo->anterior = excluir->anterior;
16             }
17         }
18
19         free(excluir);
20         return inicioLista;
21     }
22 }
```

(A) O teste da linha 7 verifica se o registro é o primeiro da lista.

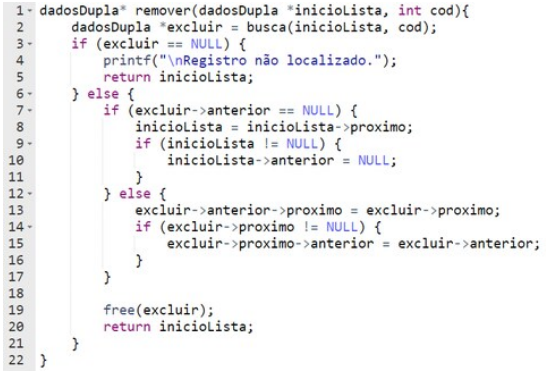
(B) Das linhas 8 à 11 serão executadas quando o registro não for encontrado na lista.

(C) Das linhas 13 à 16 serão executadas quando o registro for o primeiro da lista.

(D) A linha 18 deveria estar na linha 7, logo antes do teste do if.

(E) O retorno do inicio da lista gerará um erro de apontamento no código.

4. Sabendo que o ponteiro excluir aponta para o registro que será excluído, sobre a linha 13 do código da figura, podemos afirmar que o próximo do registro a ser excluído será enviado ao ponteiro próximo do registro anterior ao que será excluído.



```
1  dadosDupla* remover(dadosDupla *inicioLista, int cod){
2      dadosDupla *excluir = busca(inicioLista, cod);
3      if (excluir == NULL) {
4          printf("\nRegistro não localizado.");
5          return inicioLista;
6      } else {
7          if (excluir->anterior == NULL) {
8              inicioLista = inicioLista->proximo;
9              if (inicioLista != NULL) {
10                 inicioLista->anterior = NULL;
11             }
12         } else {
13             excluir->anterior->proximo = excluir->proximo;
14             if (excluir->proximo != NULL) {
15                 excluir->proximo->anterior = excluir->anterior;
16             }
17         }
18
19         free(excluir);
20         return inicioLista;
21     }
22 }
```

(V) Verdadeiro

(F) Falso