Engenharia de Telecomunicações
Programação I

19/05/2023

Recuperação Avaliação de PRG1 : PARTE 3

Professor: Eraldo Silveira e Silva

- Prova individual e sem consulta. Durante a avaliação o professor poderá monitorar a avaliação.
- usar somente ferramentas locais (não usar codespace);
- Enviar pelo SIGAA somente arquivos fontes. Compactar em arquivi único. Siga instrução do professor.

1 Exercício 1 - Valor 7

Considere um programa que possui uma tabela global de estoque de auto peças chamada *estoque_pecas[]*, onde cada item possui um identificador único, uma descrição, a quantidade em estoque e o preço da peça, conforme mostrado no esqueleto do programa abaixo. Implemente:

- uma função *retorna_item()* que recebe como parâmetros um ponteiro para uma tabela de peças, o tamanho da tabela e o identificador do item. A função retorna um ponteiro para a estrutura correspondente ao item ou NULL, caso o item não seja encontrado na tabela;
- Uma função *toral_itens()* que recebe que recebe como parâmetros um ponteiro para uma tabela de peças, o tamanho da tabela e o identificador do item. A função deve retornar o estoque de item disponíveis. Se o item não existir deve retornar -1. Esta função DEVE usar a função *retorna_item()* desenvolvida acima;
- Uma função *total_valor_estoque()* que recebe como parâmetros recebe como parâmetros um ponteiro para uma tabela de peças, o tamanho da tabela e retorna o valor total do estoque. ;
- Uma função *descricao_peca* que recebe como parâmetro um ponteiro para uma tabela de peças, o tamanho da tabela e o identificador do item e retorna um ponteiro para a descrição do tem. A função deve usar a função *retorna_item()* desenvolvida acima;
- Testar as quatro funções no *main()* de forma exaustiva. Usar como tabela global a tabela *estoque_pecas[]*.

Ver o esqueleto e saída abaixo:

```c
#include <stdio.h>
#include <string.h>

#define TAM_ID 10
#define TAM_NOME 30
#define TAM_TAB 5

typedef struct {
                    int itemID;
                    char descricao[60];
                    int quantidade;
                    float preco_unitario;
               } tipo_item;
```

```c
/* Variável Global - Tabela de Estoque de Peças */
tipo_item estoque_pecas[TAM_TAB] = {
        {355, "Coxim Lado Direito Motor", 15, 250.50},
        {850, "Coxim Lado Esquerdo Motor", 0, 700.00},
        {135, "Correia Dentada", 27, 85.00},
        {177, "Tensor Correia", 121, 55.50},
        {57, "Bomba de Agua", 0, 100.70}
};

tipo_item *retorna_item(tipo_item *ptab, int tam, int item}
{
  /* retorna um ponteiro para a estrutura que contem o item
     Deve retornar NULL se o item não está na tabela
  */
}

int total_itens (tipo_item *ptab, int tam, int item)
{
    /* retorna o total em estoque do item
       usar a função anterior para localizar o item
    */
}

float total_valor_estoque(tipo_item *ptab, int tam, int item)
{
    /* retornar o total em reais do estoque
    */
}

char *descricao_peca(tipo_item *ptab, int tam, int item)
{
    /* retornar um ponteiro para string que descreve o item
       retorna NULL se o item não existe
       esta função deve usar a função retorna_item()*/
}

int main()
{
    /*testar as funções separadamente e de forma exaustiva
      aqui.
    */
}
```

2 Exercício 2 - Valor 3

Implemente uma função C chamada *num_caracter()* **usando o conceito de ponteiros**. Ela recebe uma *string* e uma letra (char) como parâmetro e retorna o número de ocorrências da letra na *string*. Para testar esta função, entre com a *string* pela linha de comando e na sequência o caracter também na chamada do programa. O programa pode aceitar um número não limitado de strings e letres na sequência, conforme exemplo. Ver esqueleto abaixo.

```c
#include <stdio.h>


int num_caracteres(char *p, char letra)
{
  /* a função deve retornar o número caracteres iguais a letra na string  passada apontada por p */
}
```

```c
int main(int argc, char *argv[] )
{
  /* implementar os testes aqui, usando strings passadas em linha de comando
}
```

Exemplo: Suponha que o programa seja chamado no *prompt* da forma:

$ conta_caracteres abacate b batata a

A saída poderia ser:

abacate: 1 letra b
batata: 3 letras a

OBS: Note o programa deve funcionar para quaisquer número de *strings* e letras passadas. Para tanto, use o parâmetro *argc* do main().