UNIVERSIDADE TECNOLÓGICA FEDERAL DO PARANÁ

Técnicas de Programação

[AN32F-IF62C]

PROFESSOR

Diogo Cezar Teixeira Batista

LISTA DE EXERCÍCIOS - 01

ESTRUTURAS DE CONDIÇÃO, SELEÇÃO E REPETIÇÃO, VETORES, MATRIZES E FUNÇÕES

Cornélio Procópio

2017

1 INSTRUÇÕES

- Essa lista de exercícios deverá ser entregue em um arquivo único compactado no formato .zip ou .rar;
- Cada exercício deverá estar separado em um arquivo com o seguinte padrão de nomenclatura: L01_EX0N.c onde N é o número da questão;
- O trabalho é individual;
- Todo arquivo fonte deve ter como comentário o nome do aluno, da turma e o que o exercícios faz;
- Cada aluno poderá ser convocado para explicar um dos exercícios resolvidos da lista para o professor, ou para toda a turma;
- Trabalho iguais serão anulados;
- Não se esqueça de organizar, e identar o seu código;
- Utilize nomes significativos para as suas variáveis;
- Escreva as instruções para o usuário do programa que está criando;

2 DICAS

Quando se cria uma função, o estudante pode se deparar com algumas dificuldades.

1. Por onde começar?
    a. Procure ler o problema primeiro e pensar como o problema poderia ser quebrado em partes. Por exemplo, muitos problemas precisam de um programa que leia dados do usuário, faça cálculos e os apresente na tela. Então, é comum dividir o programa em uma parte de leitura, uma de cálculo e uma de apresentação. Se cada parte ainda ficar muito grande ou se essa parte gerar mais de um dado, então ela deve ser quebrada em mais partes.
    b. Uma vez que você tenha dividido o problema (e o seu futuro programa) em partes, pense como estas partes devem se relacionar.
2. Consegui pensar nas funções que meu programa terá e sei como relacioná-las. O que faço agora?

   a. Uma boa ideia é definir como cada função irá realizar sua tarefa. Por exemplo, se a função serve para ler um dado do usuário, defina as instruções de printf e scanf.
3. As funções que criei precisam de parâmetros?
   a. Uma vez que você saiba como a função deve realizar sua tarefa, você deve se perguntar: há algum dado a ser usado nesta função que foi definido fora dela? Se sim, você precisa de um parâmetro.
4. As funções que criei precisam retornar algum valor?
   a. Para responder a esta pergunta, você deve se perguntar: há algum dado definido nesta função que será usado fora dela (em outra função)? Se sim, é preciso retornar um valor.
   b. E se eu perceber que há mais de um dado definido nesta função e que serão usados em outro lugar (em outra função)? Então, você deve dividir esta função em duas (ou mais).
5. Criei cada função. Agora como faço para relacioná-las?
   a. Tudo começa na função main. A partir dela, você chamará o nome de cada função. Se a função chamada precisa de um dado externo, então você precisa passar esse valor como argumento.
   b. Se a função a ser chamada retornar um valor, então, por enquanto, sempre atribua o valor à uma variável.

3 EXERCÍCIOS

1. Faça a leitura de dois números reais na função principal, chame uma função para calcular e retornar a média entre eles, e depois imprima este resultado na função principal com duas casas de precisão;

2. Faça a leitura de dois números na função principal, depois chame uma função para verificar qual deles é maior, case o primeiro número seja maior, a função retorna o valor “1”, caso seja o segundo, a função retorna “2” e caso os dois números sejam iguais deve retornar “3”. Na função principal você deverá imprimir uma mensagem do tipo “o primeiro número é maior” ou “o segundo número é maior” ou ainda “os dois números são iguais” de acordo com o retorno da função de verificação. A função de verificação deve estar localizada abaixo da função main() no arquivo

fonte.

3. Faça a leitura dos seguintes dados no formato real: Velocidade, Tempo e Litros Gastos. Depois chame uma função para calcular e retornar o deslocamento do veículo passando como parâmetros Velocidade e Tempo. Chame outra função para calcular e retornar o consumo de combustível, passando como parâmetros Velocidade, Tempo e Litros Gastos. Esta segunda função deverá utilizar a função anterior para chegar ao consumo médio do veiculo. Imprima os resultados achados na função principal. As funções criadas deverão estar em um arquivo diferente do arquivo da função principal, este arquivo externo deverá chamar-se veiculo.c

4. Faça um programa que pergunte ao usuário qual a posição da série de Fibonacci, e depois imprima o número correspondente da série. Evite a entrada de posições inválidas (menores que 0). Para calcular o valor da série crie uma função específica para isto, que receba a posição na série e retorne o valor calculado. A série de Fibonacci tem o primeiro termo 0 igual a 0, o termo 1 igual a 1, e a partir do terceiro termo cada termo é a soma dos dois termos anteriores.

   Série de Fibonacci: *0, 1, 1, 2, 3, 5, 8, 13, ...*

5. Faça um programa que apresente o seguinte menu para o usuário.

```
Manipulação de Números
======================
<1> - Verificar se um número é par ou ímpar
<2> - Verificar se um número é primo
<3> - Calcular o fatorial de um número
<4> - Calcular a raiz quadrada de um número
<5> - Sair
======================
Qual opção escolhida?
```

   Nas opções de 1 a 4 deve-se primeiro ler o número que será usado e depois chamar uma função específica para fazer cada ação pedida. Uma função para calcular fatorial, se o número é primo, e se é par ou ímpar. Para o cálculo da raiz quadrada utilize a função sqrt da biblioteca math.h. Caso o usuário escolha a opção 5 use a função exit(int) para encerrar o programa. Caso o usuário entre com uma opção

inválida apresente uma mensagem de erro e retorne ao menu. Utilize a estrutura *switch* com *break*, e a *do/while* para fazer o menu.

6. Faça um programa que apresente o seguinte menu para o usuário:

```
Controle de Produtos
====================
<1> - Adicionar dados de um produto
<2> - Remover um produto
<3> - Listar todos os produtos
<4> - Listas apenas produtos com estoque zerado
<5> - Valor total em estoque
<6> - Sair
======================
Qual opção escolhida?
```

   O usuário poderá entrar no máximo com 30 produtos. Para cada produto armazene o código (int), a quantidade em estoque (int), e o valor unitário (float). Para isto utilize três vetores. Na opção 1 o usuário entrará com os dados de apenas um produto. Caso o usuário tente executar as opções 2, 3, 4, ou 5 e não exista nenhum produto cadastrado mostre uma mensagem de alerta. Caso o usuário tente cadastrar os dados de um novo produto e não exista mais espaço nos vetores de uma mensagem de erro. Na opção 2 o usuário deverá entrar com o código do produto que será removido, caso o produto não seja encontrado de uma mensagem de alerta, e caso o produto seja apagado corretamente de uma mensagem de sucesso. Na opção 5 deverá ser impresso o valor total em estoque, o valor em estoque de cada produto é o eu valor unitário vezes a quantidade que existe daquele produto em estoque. O valor total em estoque é a soma dos valores em estoque de todos os produtos. Caso o usuário escolha a opção 6 use a função exit(int) para encerrar o programa. Caso escolha uma opção inválida de uma mensagem de erro e retorne ao menu. As opções de 1 a 5 devem chamar funções especificas para realizar as atividades propostas. Utilize a estrutura switch com break, e a do/while para fazer o menu.

7. Faça um programa que defina uma função chamada **sumFour** que receberá por **referência** uma variável inteira, e adicionará quatro ao valor dela. Na função principal (main) crie uma variável local inteira, leia com o usuário um valor para ela, e chame a função somaQuatro passando-a. Depois imprima na tela o novo valor da

variável;

8. Faça um programa que defina uma função chamada **subTwo** que receberá por referência uma variável inteira, e subtrairá dois do valor dela, caso o novo valor não fique menor que zero. Se a função **subTwo** conseguir subtrair ela deve retornar o valor 1 indicando sucesso, caso não seja possível, a função deverá retornar 0 indicando fracasso. Na função principal (main) crie uma variável local inteira, leia com o usuário um valor para ela, e chame a função **subTwo** passando-a. Depois caso ela tenha retirado dois imprima na tela o novo valor da variável, caso não imprima uma mensagem de erro;

9. Faça um programa que leia o preço de 10 produtos e armazene em um vetor, e depois imprima a média de preços e o valor do produto mais caro. Deverá ser criada uma função para a leitura dos dados, uma que retorne a média de preços, e uma que retorne o produto mais caro. Todas estas funções devem receber o vetor como parâmetro. A função main deverá apenas declarar o vetor e chamar estas funções. Utilize constante para controlar o tamanho do vetor;

10. Faça um programa que apresente o seguinte menu para o usuário:

```
Controle de Notas
=================
<1> - Adicionar notas de um aluno
<2> - Listar todas as notas e médias
<3> - Listar a maior nota na prova 1 e na prova 2
<4> - Sair
=================
Qual opção escolhida?
```

O usuário poderá entrar no máximo com as notas de 30 alunos. Para isto crie uma matriz de reais de duas colunas, onde cada uma armazenará a nota de uma prova. Na opção 1 o usuário entrará com as duas notas de um aluno. Caso o usuário tente executar as opções 2 ou 3 e não exista nenhuma nota cadastrada mostre uma mensagem de alerta. Caso o usuário tente cadastrar os dados de um novo aluno e não exista mais espaço na matriz de uma mensagem de erro. Na opção 2 liste em formato de tabela as notas e médias de todos os alunos. Na opção 3 apresente a maior nota da prova 1 e da prova 2 cadastradas. Caso o usuário escolha a opção 4 use a função exit(int) para encerrar o programa. Caso escolha uma opção inválida

de uma mensagem de erro e retorne ao menu. As opções de 1 a 3 devem chamar funções especificas para realizar as atividades propostas, passando-se a matriz com parâmetro. Utilize a estrutura switch com break, e a do/while para fazer o menu, e constantes para controlar os limites da matriz.