UNIVERSIDADE FEDERAL DO RIO GRANDE DO NORTE
INSTITUTO METRÓPOLE DIGITAL
ITP + PTP: Arquivos, Ponteiros, Strings e Funções

Lista de Exercícios
Unidade II - 02.05.2017

Instruções Gerais

1. Esta é uma lista de exercícios para você melhorar suas habilidades em programação e, por isso, deve ser encarada com a mesma seriedade de uma avaliação;
2. Leia atentamente o enunciado de cada questão antes de iniciar a sua resposta;
3. Utilizar papel e caneta, apesar de não ser obrigatório, pode ser de grande ajuda para planejar o algoritmo que solucione os problemas. Essa prática costuma poupar tempo no momento da implementação de suas respostas.
4. Embora, nesta atividade, não sejam aplicado decréscimos de nota, lembre-se sempre e procure aplicar os mesmos padrões críticos das avaliações, evitando cometer as seguintes faltas:
   - Legibilidade comprometida (falta de indentação, etc)
   - Programa compila com mensagens de aviso (warnings)
   - Programa apresenta erros de compilação, não executa ou apresenta saída incorreta
   - Plagiarismo (no caso de cópia de colega, ambos serão penalizados)
5. **Sobre o uso de intervalos:** Um intervalo (ou *range*) consiste em um conjunto que contém todos os elementos a partir de (incluindo) *first* e antes de *last*. Dessa forma, *first* aponta para o primeiro elemento do intervalo e *last* aponta para o primeiro elemento APÓS o intervalo (intervalo fechado-aberto). Esse modelo de definição de intervalo [first,last) é usado com frequência por bibliotecas profissionais, e também **DEVE** ser utilizado na resolução das questões abaixo.

Questão 1. Escreva uma função **negate()** que inverte o sinal de um dado intervalo de inteiros. O programa deve ler n valores inteiros e armazena-los em um vetor. Este vetor deverá ser submetido à função **negate()**, através dos ponteiros first e last e depois os valores de seus elementos deverão ser impressos. A função deve possuir o seguinte protótipo (ou assinatura):

```
void negate ( int * first, int * last );
```

Escreva um programa para testar a sua função.

Exemplo de Entrada	Exemplo de Saída
8 2 4 7 -3 9 0 0 15	-2 -4 -7 3 -9 0 0 -15

Questão 2. Escreva uma função **scalar()** que multiplica um intervalo de inteiros por um outro número inteiro. O programa deve ler o número n de valores inteiros a serem lidos, seguido do valor escalar m pelo qual deverá multiplicar cada um dos n valores inteiros a serem lidos a seguir.

```
void scalar ( int * first, int * last , int m);
```

*s*Implemente um programa para testar a sua função.

Exemplo de Entrada	Exemplo de Saída
8 3 2 4 7 -3 9 0 0 15	6 12 21 -9 27 0 0 45

Questão 3. Escreva a função **swap()** que troca **o valor** de duas variáveis do tipo inteiro. A função deve possuir o seguinte protótipo (ou assinatura):

```
void swap(int* x, int* y);
```

Implemente um programa que teste a sua função.

Questão 4. Escreva um programa para classificar o peso de uma pessoa, dados seu peso e altura. A classificação é baseada no IMC (Índice de Massa Corporal). A fórmula para o cálculo do IMC e as faixas de classificação são dadas abaixo:

$$IMC = \frac{PESO(Kg)}{ALTURA(m)^2} \tag{1}$$

IMC	Classificação
Abaixo de 17	Muito abaixo do peso
Entre 17 e 18,49	Abaixo do peso
Entre 18,5 e 24,99	Peso normal
Entre 25 e 29,99	Acima do peso
Entre 30 e 34,99	Obesidade I
Entre 35 e 39,99	Obesidade II (severa)
Acima de 40	Obesidade III (mórbida)

O cálculo do IMC deve ser realizado em uma função específica.

Questão 5. A operadora de energia de uma cidade aplica, para cada casa, o valor básico de R$ 20,00, para um consumo mensal de até 45 Kwh. Para cada Kwh excedente é cobrado o valor de R$ 0,50. Escreva um programa para ler a quantidade de Kwh consumida no mês por uma casa e depois mostrar o valor da conta de energia. Utilize uma função para o cálculo do valor da conta, dado o consumo.

Questão 6. Desenvolva um programa **compacta** que elimina todos os elementos menores ou iguais a zero do intervalo [first,last) e imprime o vetor resultante. A ordem relativa entre os elementos deve ser preservada. Considere o exemplo abaixo com apenas 10 elementos.

Vetor Não-Compactado	Vetor Compactado
-2 -8 2 7 -3 10 1 0 -3 7	2 7 10 1 7

Perceba, no entanto, que os outros elementos do vetor não são modificados. Portanto, ao tentar imprimir o vetor original completo, obtemos a seguinte saída:

Vetor Completo
2 7 10 1 7 10 1 0 -3 7

A função deve possuir o seguinte protótipo (ou assinatura):

```
void compacta ( int * first, int * last );
```

Seu programa deve receber um número inteiro n, depois deve ler n números inteiros, realizar a compactação, imprimir o vetor compactado, e imprimir o vetor completo.

Exemplo de Entrada	Exemplo de Saída
10 -2 -8 2 7 -3 10 1 0 -3 7	2 7 10 1 7 2 7 10 1 7 10 1 0 -3 7

Questão 7. Considere que os elementos no intervalo [first,last) podem ser classificados como de cor PRETA (valor 1) ou BRANCA (valor 0) e estão dispostas em uma ordem qualquer. Desenvolva um programa **sort-marbles** que rearranja os elementos do intervalo de maneira que todas as ocorrências PRETAs apareçam antes de todas ocorrências BRANCAs, e imprime um inteiro que corresponde ao limite das regiões ou seja, indicando a posição do primeiro elemento BRANCO.

Figura 1: Rearranjando um vetor (à esquerda) de maneira a separar região PRETA e BRANCA (à direita). Neste exemplo o algoritmo deveria imprimir o número 3, assumindo que o intervalo compreende todo o vetor (i.e. a partir de zero).

Seu programa deve receber um número n, uma sequência de n números inteiros 0s ou 1s (com 0s representando BRANCO e 1s representando PRETO), imprimir a sequência ordenada (brancas primeiro) e logo após o número m representando o limite entre as regiões branca e preta.

Exemplo de Entrada	Exemplo de Saída
7 0 1 1 0 0 1 0	1 1 1 0 0 0 0 3

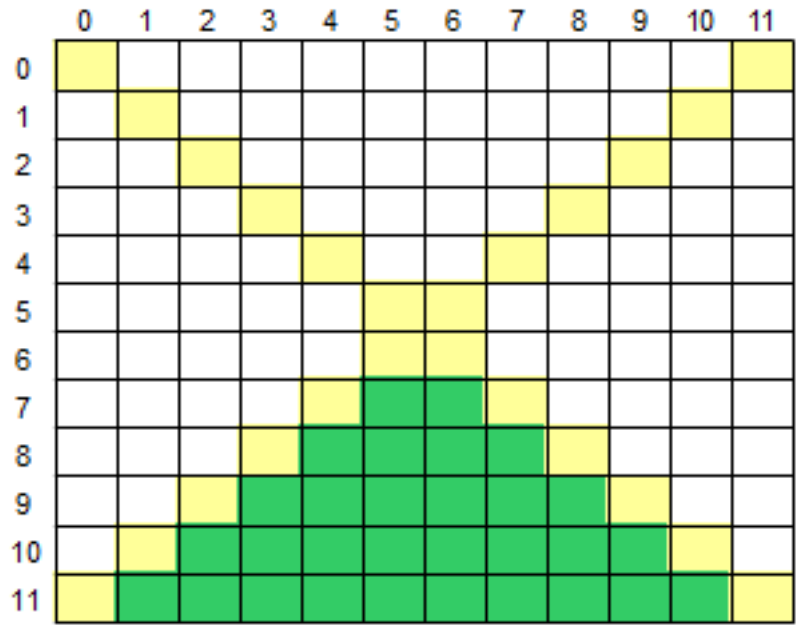
Questão 8. Elabore um programa que leia valores inteiros de um arquivo e armazene em um vetor de inteiros de N posições, usando a função load(). Após preencher o vetor, deverão ser criados dois indices para o vetor: um em ordem crescente de valores e o outro em ordem decrescente. Um índice é um vetor de ponteiros de inteiros de mesmo tamanho N, onde os elementos do índice apontam para o vetor de inteiros na ordem definida para o índice. Crie ainda as funções print() e save(). A primeira imprime os elementos do vetor na ordem definida pelo índice passado por parâmetro, enquanto a segunda salva os valores do vetor de inteiros em um arquivo e seguindo a ordem imposta pelo índice passado por parâmetro.

Utilize as seguintes assinaturas para as funções:

```
void load (int* vetor, int* tamanho, const char* infile);
int** createIdxCrescente (const int* vetor, const int tamanho);
int** createIdxDecrescente (const int* vetor, const int tamanho);
void print (const int* indice, const int tamanho);
void save (const int* indice, const int tamanho, const char* outfile);
```

Por fim, implemente um programa de teste.

Questão 9. (URI Online Judge | 1188) **Área Inferior**. Leia um caractere maiúsculo, que indica uma operação que deve ser realizada e uma matriz M[12][12]. Em seguida, calcule e mostre a soma ou a média considerando somente aqueles elementos que estão na área inferior da matriz, conforme ilustrado abaixo (área verde).



Entrada

A primeira linha de entrada contem um único caractere Maiúsculo O ('S' ou 'M'), indicando a operação (Soma ou Média) que deverá ser realizada com os elementos da matriz. Seguem os 144 valores de ponto flutuante de dupla precisão (double) que compõem a matriz.

Saída

Imprima o resultado solicitado (a soma ou média), com 1 casa após o ponto decimal.

Exemplo de Entrada	Exemplo de Saída
S 1.0 330.0 -3.5 2.5 4.1 ...	112.4

Questão 10. Números amigos são dois números que estão ligados um ao outro por uma propriedade especial: cada um deles é a soma dos divisores do outro (Wikipedia). Os divisores próprios de um número positivo N são todos os divisores inteiros positivos de N exceto o próprio N.

Um exemplo conhecido de números amigos são 284 e 220, pois os divisores próprios de 220 são 1, 2, 4, 5, 10, 11, 20, 22, 44, 55 e 110. Efetuando a soma destes números obtemos o resultado 284.

Exemplo de Numeros Amigos

220: 1+2+4+5+10+11+20+22+44+55+110=284
284: 1+2+4+71+142=220

A descoberta deste par de números é atribuída à Pitágoras.

Houve uma aura mística em torno deste par de números, e estes representaram papel importante na magia, feitiçaria, na astrologia e na determinação de horóscopos.

Outros números amigos foram descobertos com o passar do tempo. Pierre Fermat anunciou em 1636 um novo par de números amigos formando por 17296 e 18416, mas na verdade tratou-se de uma redescoberta pois o árabe al-Banna (1256 - 1321) já havia encontrado este par de números no fim do século XIII.

Leonardo Euler, matemático suíço, estudou sistematicamente os números amigos e descobriu em 1747 uma lista de trinta pares, e ampliada por ele mais tarde para mais de sessenta pares. Todos os números amigos inferiores a um bilhão já foram encontrados (Fonte[1]).

Implemente um programa em C que encontre todos os pares de números inteiros amigos até 1 milhão. Organize seu código com o uso de funções.

Exemplo de Saída

(220, 284)
(1184, 1210)
(2620, 2924)
(5020, 5564)
(6232, 6368)
(10744, 10856)
...
(898216, 980984)

Questão 11. Escreva uma função que aceita como parâmetro um vetor de inteiros com N valores, e determina o maior elemento do vetor e o numero de vezes que este elemento ocorreu no vetor. Por exemplo, para um vetor com os seguintes elementos: [5, 2, 15, 3, 7, 15, 8, 6, 15] a função deve retornar para o programa que a chamou o valor 15 e o número 3, indicando que o numero 15 ocorreu 3 vezes. A funcão deve ser do tipo void.

[1] http://www.matematica.br/historia/namigos.html

Exemplo de Entrada	Exemplo de Saída
9 5 2 15 3 7 15 8 6 15	O maior valor e 15 e ocorre 3 vezes

Questão 12. Considere a seguinte declaração:

```
1  int a,*b,**c,***d;
```

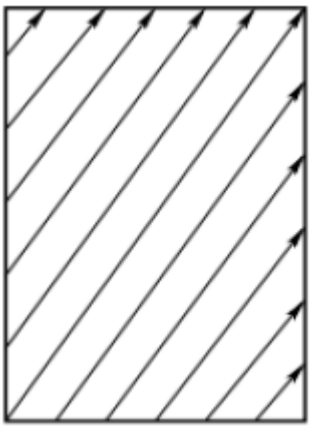
Escreva um programa que leia o valor da variável **a**, calcule e exiba o dobro, o triplo e o quádruplo desse valor utilizando apenas os ponteiros b, c e d. Não devem ser usadas outras variáveis ou constantes, apenas manipulação de ponteiros. O ponteiro b deve ser usada para calcular o dobro, c para calcular o triplo e d para calcular o quadruplo.

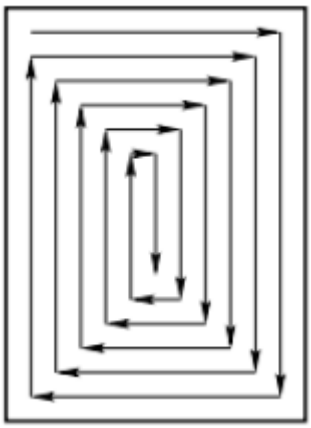
Exemplo de Entrada	Exemplo de Saída
3	Valor: 3 Dobro: 6 Triplo: 9 Quadruplo: 12

Questão 13. Hora de mostrar seu domínio em ponteiros. Qual a saída do programa abaixo? Não basta dizer o que, pois esta parte é fácil (basta executar o código! ;P). Tem que explicar como se dá a "Magia Negra!". Utilize os conceitos e terminologias discutidas em sala de aula.

```
1  #include <stdio.h>
2
3  #define N 3
4
5  int main(int argc, char const *argv[])
6  {
7    int numeros[N] = {1768382797, 1699618913, 560034407};
8    char* p;
9    p = (char*) numeros;
10   for (int i = 0; i < N*4; ++i)
11    {
12     printf("%c", *(p++));
13    }
14   return 0;
15 }
```

Questão 14. Crie 2 funções em C que permitam percorrer uma matriz bidimensional seguindo os seguintes formatos.





Seu programa não deve impor limitações sobre o número de linhas, nem colunas. Não use a notação de colchetes. Use ponteiros.

Apresente um programa de teste e exemplos (quanto mais melhor), mostrando que a sua implementação funciona. Seu programa deve ser claro e bem indentado.

Exemplo de Entrada

```
7 4
1 2 3 4
5 6 7 8
9 10 11 12
13 14 15 16
17 18 19 20
21 22 23 24
25 26 27 28
```

Exemplo de Saída

```
PERCURSO EM ESPIRAL:
1 2 3 4 8 12 16 20 24 28 27 26 25 21 17
13 9 5 6 7 11 15 19 23 22 18 14 10

PERCURSO EM DIAGONAL:
1 5 2 9 6 3 13 10 7 4 17 14 11 8 21 18 15
12 25 22 19 16 26 23 20 27 24 28
```

Questão 15. Implemente um programa no qual o usuario informa, através da linha de comando, o nome de um arquivo de entrada e uma palavra, e o programa lê o arquivo em busca das ocorrências da palavra. Ao final, o programa imprime o numero de vezes que aquela palavra aparece no arquivo.

Questão 16. Implemente uma função que receba duas strings e verifique se a segunda string esta contida na primeira, retornando como resultado a posição de início da segunda palavra na primeira. Caso a segunda palavra não ocorra na primeira, retornar -1. Implemente tambem um programa que teste a sua função.

```
1 int a,*b,**c,***d;
```

Exemplo de Entrada

```
111
```

Exemplo de Saída

```
111
```

Questão 17. Implemente um programa no qual, dado um número N, retorne a soma dos algarismos de N!. Ex: se N = 4, N! = 24. Logo, a soma de seus algarismos é 2 + 4 = 6. Organize o seu programa em funções. Apresente alguns casos de teste.

Questão 18. Implemente um programa que lê um inteiro X e retorna o maior número inteiro primo anterior ao valor do fatorial de X. Organize o seu programa com funções.

Por exemplo, se X = 5, temos que fatorial de X = 120, logo o maior número inteiro primo anterior a X é 113.

Exemplo de Entrada	Exemplo de Saída
5 3 9	113 5 362867