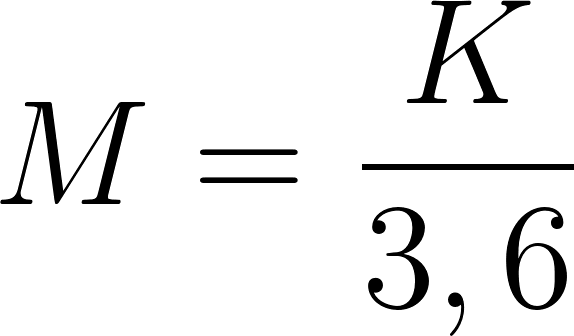
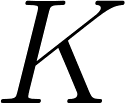
# Aula Prática 4

**Prazo de entrega**: conferir no Moodle e no cronograma.

**Forma de Entrega:** todas as funções abaixo devem ser implementadas em módulo pratica4 (com os arquivos pratica4.c e pratica4.h). Todos os programas devem ser implementados em um arquivo .c separado (exemplo: pratica4ex15.c). Crie também um arquivo .c separado para testar as funções que não foram testadas (testePratica4.c). Compacte todos os arquivos .c e .h implementados em um zip e o envie pelo Moodle.

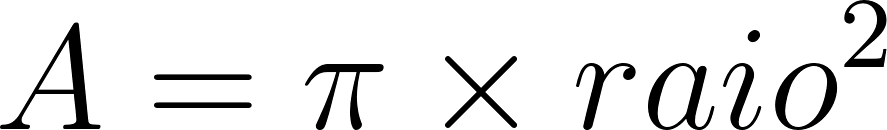
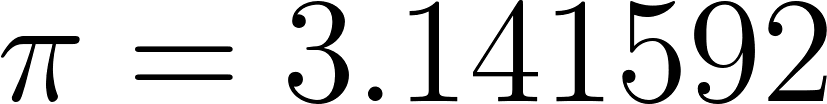
**Exercícios de processamento condicional:**

**Exercício 1**

Implementar uma função que recebe como parâmetro uma velocidade em km/h (quilômetros por hora) e retorne a mesma convertida para m/s (metros por segundo). A fórmula de conversão é [](https://www.codecogs.com/eqnedit.php?latex=M%20%3D%20%5Cfrac%7BK%7D%7B3%2C6%7D%0), sendo [](https://www.codecogs.com/eqnedit.php?latex=K%0) a velocidade em km/h e [](https://www.codecogs.com/eqnedit.php?latex=M%0) a velocidade em m/s. Protótipo:

float paraMetrosPorSegundo(float v);

**Exercício 2**

Implementar uma função que recebe como parâmetro o raio de um círculo e retorne a área do círculo correspondente. A área do círculo é [](https://www.codecogs.com/eqnedit.php?latex=A%20%3D%20%5Cpi%20%5Ctimes%20raio%5E2%0), sendo que [](https://www.codecogs.com/eqnedit.php?latex=%5Cpi%20%3D%203.141592%0). Protótipo:

float areaCirculo(float raio);

**Exercício 3**

Implementar uma função que recebe três números inteiros como parâmetro e retorna o maior entre eles. Protótipo:

int maior3(int n1, int n2, int n3);

**Exercício 4**

Implementar uma função que recebe como parâmetro um número inteiro n e retorne 1 se ele for par e 0 caso ele seja ímpar. Protótipo:

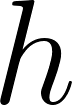
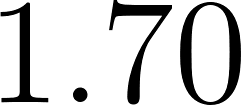
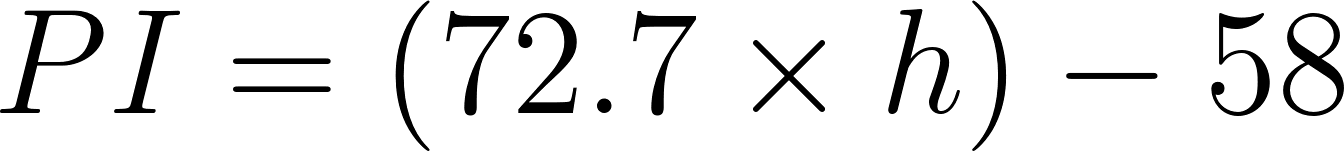
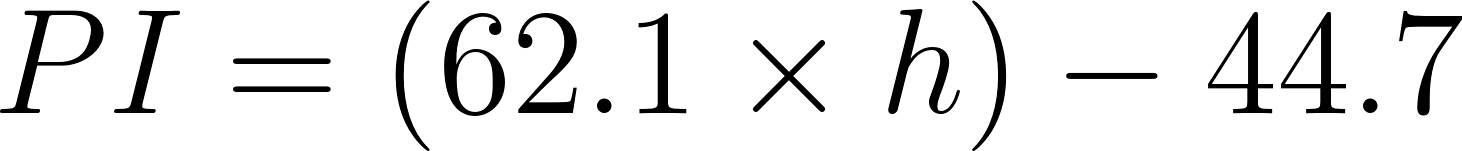
int ehPar(int n);

**Exercício 5**

Implementar uma função que recebe como parâmetro um número inteiro n e retorne 1 se ele for divisível por 3 ou por 5, mas não simultaneamente pelos dois, ou 0 caso contrário (divisível por 3 e 5 ou por nenhum dos dois). Protótipo:

int ehDivisivelPor3ou5(int n);

**Exercício 6**

Implemente uma função que recebe como parâmetro a altura [](https://www.codecogs.com/eqnedit.php?latex=h%0) em metros (exemplo: [](https://www.codecogs.com/eqnedit.php?latex=1.70%0)) e o sexo ('M' para masculino e 'F' para feminino) de uma pessoa e retorne o seu peso ideal [](https://www.codecogs.com/eqnedit.php?latex=PI%0), sendo que [](https://www.codecogs.com/eqnedit.php?latex=PI%3D(72.7%5Ctimes%20h)%20-%2058%0) caso o sexo seja masculino e [](https://www.codecogs.com/eqnedit.php?latex=PI%3D(62.1%5Ctimes%20h)%20-%2044.7%0) caso feminino. Protótipo:

float pesoIdeal(float h, char sexo);

**Exercício 7**

Implemente um programa para ler o sexo, a altura, e o peso do usuário e informar quantos quilogramas ele deve ganhar ou perder para alcançar o seu peso ideal. Use a função pesoIdeal do exercício anterior.

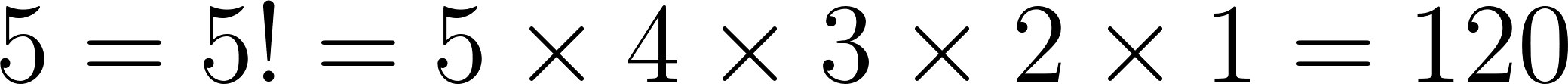
**Exercícios de processamento iterativo:**

**Exercício 8**

Implementar uma função que recebe como parâmetro um número inteiro N e retorne a soma dos números ímpares de 0 até N (incluindo N, se N for ímpar). Protótipo:

int somaImpares(int N);

**Exercício 9**

Implementar uma função que recebe como parâmetro um número inteiro [](https://www.codecogs.com/eqnedit.php?latex=N%0) e retorne o seu fatorial. Exemplo: o fatorial de [](https://www.codecogs.com/eqnedit.php?latex=5%20%3D%205!%20%3D%205%5Ctimes%204%5Ctimes%203%5Ctimes%202%5Ctimes%201%20%3D%20120%0). Protótipo:

double fatorial(int N);

**Exercício 10**

Implementar uma função que recebe como parâmetro um número inteiro N e retorne a soma de todos os números positivos menores ou iguais a N que são divisíveis por 3 ou por 5, mas não por ambos. Exemplo: para N==20, a soma é 3+5+6+9+10+12+18+20 = 83. Protótipo:

int somaNumerosDiv3ou5(int N);

**Exercício 11**

Implemente uma função que recebe como parâmetro três números inteiros maiores que zero (não precisa testar) x,y,z e uma operação numérica que pode assumir os valores 1,2,3 e 4. Caso a operação seja 1, a função deve calcular a média geométrica, caso seja 2, a média ponderada, caso seja 3, a média harmônica e, por fim, caso seja 4, a média aritmética. Confira a tabela abaixo para as fórmulas. Protótipo:

float calculaMedia(int x, int y, int z, int operacao);

| **Operação** | **Média** | **Fórmula** |
| --- | --- | --- |
| 1 | Geométrica |  |
| 2 | Ponderada |  |
| 3 | Harmônica |  |
| 4 | Aritmética |  |

**Exercício 12**

Implementar uma função que recebe como parâmetro um número inteiro N e retorne o seu número de divisores. Exemplo: os divisores de 66 são 8: 1,2,3,6,11,22,33,66. Protótipo:

int numeroDivisores(int N);

**Exercício 13**

Implementar uma função que recebe como parâmetro um número inteiro positivo N e retorne o enésimo termo da sequência de Fibonacci. Essa sequência começa no termo de ordem zero e, a partir do segundo termo, seu valor é dado pela soma dos dois termos anteriores. Exemplo: para N == 8, o enésimo termo é 13, uma vez que a sequência de Fibonacci até o oitavo termo é: 0,1,1,2,3,5,8,13. Protótipo:

int enesimoFibonacci(int N);

**Exercício 14**

Escreva uma *função* que retorna o máximo divisor comum (MDC) entre dois números inteiros. O MDC entre dois números é o maior número inteiro que os divide. Ex: O MDC entre 18 e 12 é 6. O MDC entre 18 e 6 é 6. Essa função deve ter o seguinte protótipo:

int mdc(unsigned int x, unsigned int y);

**Exercício 15**

Escreva uma função que retorna o mínimo múltiplo comum (MMC) entre dois números inteiros. O MMC entre dois números é o menor número inteiro que é múltiplo de ambos, ou seja, que tem como divisores os dois números. Ex: O MMC entre 18 e 12 é 36. O MMC entre 18 e 6 é 18. Essa função deve ter o seguinte protótipo:

int mmc(unsigned int x, unsigned int y);

**Exercício 16**

Escreva um *programa* que lê dois números inteiros do teclado e imprime na tela o máximo divisor comum (MDC) e o mínimo múltiplo comum (MMC) entre eles. Caso o usuário insira qualquer valor menor ou igual a zero, o programa deve informar isso a ele e pedir um novo número. Esse processo deve se repetir enquanto qualquer um dos números lidos seja menor ou igual a zero.