Exercícios de IF, ELIF e ELSE

1. Faça um Programa que verifique se uma letra digitada é vogal ou consoante.

2. Faça um programa que pede duas notas de um aluno. Em seguida ele deve calcular a média do aluno e dar o seguinte resultado:

A mensagem "Aprovado", se a média alcançada for maior ou igual a sete;
A mensagem "Reprovado", se a média for menor do que sete;
A mensagem "Aprovado com Distinção", se a média for igual a dez.

3. Faça um Programa que leia três números inteiros e mostre o maior deles.

4. Faça um Programa que leia três números inteiros, em seguida mostre o maior e o menor deles.

5. Faça um programa que pede dois inteiro e armazene em duas variáveis. Em seguida, troque o valor das variáveis e exiba na tela

6. Faça um Programa que leia três números e mostre-os em ordem decrescente.

7. Faça um Programa que pergunte em que turno você estuda. Peça para digitar M-matutino ou V-Vespertino ou N- Noturno. Imprima a mensagem "Bom Dia!", "Boa Tarde!" ou "Boa Noite!" ou "Valor Inválido!", conforme o caso.

8. As Organizações Tabajara resolveram dar um aumento de salário aos seus colaboradores e lhe contrataram para desenvolver o programa que calculará os reajustes.

Faça um programa que recebe o salário de um colaborador e o reajuste segundo o seguinte critério, baseado no salário atual:

salários até R$ 280,00 (incluindo) : aumento de 20%
salários entre R$ 280,00 e R$ 700,00 : aumento de 15%
salários entre R$ 700,00 e R$ 1500,00 : aumento de 10%
salários de R$ 1500,00 em diante : aumento de 5% Após o aumento ser realizado, informe na tela:

o salário antes do reajuste;
o percentual de aumento aplicado;
o valor do aumento;
o novo salário, após o aumento.

9. Faça um programa para o cálculo de uma folha de pagamento, sabendo que os descontos são do Imposto de Renda, que depende do salário bruto (conforme tabela abaixo) e 3% para o Sindicato e que o FGTS corresponde a 11% do Salário Bruto, mas não é descontado (é a empresa que deposita). O Salário Líquido corresponde ao Salário Bruto menos os descontos. O programa deverá pedir ao usuário o valor da sua hora e a quantidade de horas trabalhadas no mês.

Desconto do IR:
Salário Bruto até 900 (inclusive) - isento
Salário Bruto até 1500 (inclusive) - desconto de 5%
Salário Bruto até 2500 (inclusive) - desconto de 10%
Salário Bruto acima de 2500 - desconto de 20% Imprima na tela as informações, dispostas conforme o exemplo abaixo. No exemplo o valor da hora é 5 e a quantidade de hora é 220.

Salário Bruto: (5 * 220) : R$ 1100,00
(-) IR (5%) : R$ 55,00
(-) INSS (10%) : R$ 110,00
FGTS (11%) : R$ 121,00
Total de descontos : R$ 165,00
Salário Liquido : R$ 935,00

10. Faça um Programa que leia um número e exiba o dia correspondente da semana. (1-Domingo, 2- Segunda, etc.), se digitar outro valor deve aparecer valor inválido.

11.Faça um programa que lê as duas notas parciais obtidas por um aluno numa disciplina ao longo de um semestre, e calcule a sua média. A atribuição de conceitos obedece à tabela abaixo:

Média de Aproveitamento Conceito

Entre 9.0 e 10.0	A
Entre 7.5 e 9.0	B
Entre 6.0 e 7.5	C
Entre 4.0 e 6.0	D
Entre 4.0 e zero	E

O algoritmo deve mostrar na tela as notas, a média, o conceito correspondente e a mensagem “APROVADO” se o conceito for A, B ou C ou “REPROVADO” se o conceito for D ou E.

12. Faça um Programa que peça os 3 lados de um triângulo. O programa deverá informar se os valores podem ser um triângulo. Indique, caso os lados formem um triângulo, se o mesmo é: equilátero, isósceles ou escaleno.

Dicas:
Três lados formam um triângulo quando a soma de quaisquer dois lados for maior que o terceiro;
Triângulo Equilátero: três lados iguais;
Triângulo Isósceles: quaisquer dois lados iguais;
Triângulo Escaleno: três lados diferentes;

13. Faça um programa que calcule as raízes de uma equação do segundo grau, na forma $ax^2 + bx + c$. O programa deverá pedir os valores de a, b e c e fazer as consistências, informando ao usuário nas seguintes situações:

Se o usuário informar o valor de A igual a zero, a equação não é do segundo grau e o programa não deve fazer pedir os demais valores, sendo encerrado;
Se o delta calculado for negativo, a equação não possui raizes reais. Informe ao usuário e encerre o programa;
Se o delta calculado for igual a zero a equação possui apenas uma raiz real; informe-a ao usuário;
Se o delta for positivo, a equação possui duas raiz reais; informe-as ao usuário;

PS: digite 'import math' no início de seu script. Para achar a raiz quadrada da variável x, faça: math.sqrt(x)

14. Faça um Programa que peça um número correspondente a um

determinado ano e em seguida informe se este ano é ou não bissexto.

15. Faça um Programa que peça uma data no formato dd/mm/aaaa e determine se a mesma é uma data válida.

16. Faça um Programa que peça um número inteiro e determine se ele é par ou ímpar. Dica: utilize o operador módulo (resto da divisão): %

17. Faça um Programa que leia um número inteiro menor que 1000 e imprima a quantidade de centenas, dezenas e unidades do mesmo. Observando os termos no plural a colocação do "e", da vírgula entre outros. Exemplo:

326 = 3 centenas, 2 dezenas e 6 unidades
12 = 1 dezena e 2 unidades Testar com: 326, 300, 100, 320, 310,305, 301, 101, 311, 111, 25, 20, 10, 21, 11, 1, 7 e 1

18. Faça um Programa para um caixa eletrônico. O programa deverá perguntar ao usuário a valor do saque e depois informar quantas notas de cada valor serão fornecidas. As notas disponíveis serão as de 1, 5, 10, 50 e 100 reais. O valor mínimo é de 10 reais e o máximo de 600 reais. O programa não deve se preocupar com a quantidade de notas existentes na máquina.

Exemplo 1: Para sacar a quantia de 256 reais, o programa fornece duas notas de 100, uma nota de 50, uma nota de 5 e uma nota de 1;
Exemplo 2: Para sacar a quantia de 399 reais, o programa fornece três notas de 100, uma nota de 50, quatro notas de 10, uma nota de 5 e quatro notas de 1.

19.Faça um Programa que peça um número e informe se o número é inteiro ou decimal. Dica: utilize uma função de arredondamento.

20. Faça um Programa que leia 2 números e em seguida pergunte ao usuário qual operação ele deseja realizar. O resultado da operação deve ser acompanhado de uma frase que diga se o número é:

par ou ímpar;
positivo ou negativo;

inteiro ou decimal.

21. Faça um programa que faça 5 perguntas para uma pessoa sobre um crime. As perguntas são:
"Telefonou para a vítima?"
"Esteve no local do crime?"
"Mora perto da vítima?"
"Devia para a vítima?"
"Já trabalhou com a vítima?" O programa deve no final emitir uma classificação sobre a participação da pessoa no crime. Se a pessoa responder positivamente a 2 questões ela deve ser classificada como "Suspeita", entre 3 e 4 como "Cúmplice" e 5 como "Assassino". Caso contrário, ele será classificado como "Inocente".

22. Um posto está vendendo combustíveis com a seguinte tabela de descontos:
Álcool: até 20 litros, desconto de 3% por litro
acima de 20 litros, desconto de 5% por litro

Gasolina:
até 20 litros, desconto de 4% por litro
acima de 20 litros, desconto de 6% por litro Escreva um algoritmo que leia o número de litros vendidos, o tipo de combustível (codificado da seguinte forma: A-álcool, G-gasolina), calcule e imprima o valor a ser pago pelo cliente sabendo-se que o preço do litro da gasolina é R$ 2,50 o preço do litro do álcool é R$ 1,90.

23. Uma fruteira está vendendo frutas com a seguinte tabela de preços:

	Até 5 Kg	Acima de 5 Kg
Morango	R$ 2,50 por Kg	R$ 2,20 por Kg
Maçã	R$ 1,80 por Kg	R$ 1,50 por Kg

Se o cliente comprar mais de 8 Kg em frutas ou o valor total da compra ultrapassar R$ 25,00, receberá ainda um desconto de 10% sobre este total. Escreva um algoritmo para ler a quantidade (em Kg) de morangos e a quantidade (em Kg) de maças adquiridas e escreva o valor a ser pago pelo cliente.

24. O Hipermercado Tabajara está com uma promoção de carnes que é imperdível. Confira:

Até 5 Kg		Acima de 5 Kg
File Duplo	R$ 4,90 por Kg	R$ 5,80 por Kg
Alcatra	R$ 5,90 por Kg	R$ 6,80 por Kg
Picanha	R$ 6,90 por Kg	R$ 7,80 por Kg

Para atender a todos os clientes, cada cliente poderá levar apenas um dos tipos de carne da promoção, porém não há limites para a quantidade de carne por cliente. Se compra for feita no cartão Tabajara o cliente receberá ainda um desconto de 5% sobre o total a compra. Escreva um programa que peça o tipo e a quantidade de carne comprada pelo usuário e gere um cupom fiscal, contendo as informações da compra: tipo e quantidade de carne, preço total, tipo de pagamento, valor do desconto e valor a pagar.

Soluções

1.

Primeiro, pedimos um caractere ao usuário e guardamos na variável 'caractere'.

O normal agora seria comparamos se é uma vogal 'a', 'e', 'i', 'o' ou 'u' ou não. Nós vamos fazer isso sim, porém tem uma pegadinha aí.

Em programação, uma letra minúscula é diferente da sua maiúscula.
Logo, além das vogais que citamos, temos que comparar com 'A', 'E', 'I', 'O' e 'U' também.

Nosso código fica:

```
char=input('Digite um caractere: ')

if char=='a' or char=='e' or char=='i' or char=='o' or char=='u' or \
   char=='A' or char=='E' or char=='I' or char=='O' or char=='U':
      print('Vogal')
else:
      print('Consoante')
```

Para um if não ficar muito grande, com muitas condições, quebramos ele em duas linhas. Para isso, basta usar uma barra \ , como fizemos no código.

2.

Pedimos as notas ao usuário e armazenamos nas variáveis 'nota1' e 'nota2'.
Não esqueça de transformá-las em decimal, usando a função float()

Em seguida, calculamos a média.
Use parêntesis para não ter problema de precedência de operadores.

Agora vamos aos testes.
É sempre interessante tratar logo a condição mais simples e que exclui logo todo o restante.

Nesse caso, testamos logo se a média é menor que 7.
Se for, diz reprovado e encerra o script.

Caso não seja menor, é porque é maior ou igual a 7.

Aqui temos que fazer outro teste: já sabemos que é 7.0 ou mais, mas esse número é menor 10.0 ?
Se for, então exibe a mensagem de parabéns.

Se não for menor que 10.0, é porque é 10.0 cravado, então exibimos a mensagem de aprovação com distinção.

```
    nota1 = float(input('Primeira nota: '))
    nota2 = float(input('Segunda nota: '))

    media = (nota1 + nota2) / 2

    print('Media: ',media)

    if media<7.0:
        print('Reprovado')
    elif media<10:
        print('Aprovado')
    else:
        print('Aprovado com Distinção!')
```

3.

Vamos armazenar os três números que o usuário digitar nas variáveis 'primeiro', 'segundo' e 'terceiro'.

O pulo do gato, nessa questão, é usar uma variável extra, que chamaremos de 'maior'. A função dessa variável é simples: armazenar o maior valor que ela achar.

Inicialmente, fazemos com que 'maior' aponte para a variável 'primeiro', estamos supondo que o número 'primeiro' é o maior:
maior = primeiro

Agora vamos fazer os testes!
Vamos comparar o primeiro número com o segundo.
Se o segundo for maior que o primeiro, a variável 'maior' agora vai receber o valor da variável 'segundo':
maior = segundo

Se não for maior, então 'maior' ainda está com o valor da variável 'primeira', que definimos no começo. Então a variável 'maior' vai ter sempre o maior valor, entre os dois primeiros números digitados, concorda?

Agora vamos testar se a variável 'terceiro' é maior que o valor 'maior'.

Se for, o novo valor de 'maior' vai ser o terceiro número:
maior = terceiro

Se não for maior que 'terceiro', ela continua com valor anterior e este é o maior de todos.

Nosso código fica:

```
    primeiro = int(input('Primeiro numero: '))
    segundo  = int(input('Segundo numero : '))
    terceiro = int(input('Terceiro numero: '))

    maior = primeiro

    if (segundo > maior):
        maior = segundo
    if (terceiro > maior):
        maior = terceiro

    print('Maior: ',maior)
```

Não se assuste se não entender de cara.
Leia, releia, pense, pense de novo, reflita, chore em posição fetal até entender.

Programar é assim mesmo. O segredo é continuar tentando entender, continuar pensando…

4.

Vamos usar o mesmo código da questão anterior, pra achar o maior número:
Achar o maior número em Python

Em seguida, vamos achar o menor.
A lógica é a mesma de achar o maior, primeiro fazemos com que a variável 'menor' receba o valor do 'primeiro' número.

Em seguida, comparamos se o segundo valor é menor que o primeiro.
Se for, a variável 'menor' agora vai receber o valor de 'segundo'.
Se não for menor, fica como está ('menor' continua com o valor de 'primeiro', que é o menor entre os dois primeiros números).

Agora vamos fazer o mesmo teste com o terceiro número.

Se ele for menor que o valor armazenado em 'menor', fazemos com que 'menor' receba o valor de 'terceiro'.

Nosso código fica:

```
primeiro = int(input('Primeiro numero: '))
   segundo  = int(input('Segundo numero : '))
   terceiro = int(input('Terceiro numero: '))

   # Achando o maior número
   maior = primeiro

   if (segundo > maior):
      maior = segundo
   if (terceiro > maior):
      maior = terceiro

   print('Maior: ',maior)

   # Achando o menor número
   menor = primeiro

   if (segundo < menor):
      menor = segundo
   if (terceiro < menor):
      menor = terceiro

   print('Menor: ',menor)
```

5.

Temos duas variáveis: var1 e var2

Vamos fazer com que **var2** receba o valor de **var1**:
var2 = var1

Agora vamos fazer com que **var1** receba o valor de **var2**...epa!
Vai dar erro, pois eu MUDEI o valor de **var2** no comando anterior!
O valor originalmente em **var2** foi perdido quando fiz essa variável mudar de valor.

E agora, José?
Calma, você faz o **Curso Python Progressivo**, e vai aprender a resolver isso.

O segredo é: usar uma variável auxiliar, a **aux**

A função dar **aux** é guardar aquele primeiro valor contido em **var2.**
Então, a troca de valores se dá assim:

aux = var2
var2 = var1
var1 = aux

Faz sentido pra você?
Reflita e veja se entender **perfeitamente**, pois esse algoritmo de troca é MUITO importante!

```
var1 = int(input('Primeiro numero: '))
var2  = int(input('Segundo numero : '))

print('Variavel 1: ',var1)
print('Variavel 2: ',var2)
print('Invertendo...')

aux  = var2
var2 = var1
var1 = aux

print('Variavel 1: ',var1)
print('Variavel 2: ',var2)
```

6.

O grande segredo desse tipo de algoritmo, é comparar e trocar valor das variáveis, duas a duas.

Por exemplo, vamos pegar a sequência: 10 - 20 - 30
Primeira posição: 10
Segunda posição: 20
Terceira posição: 30

Primeiro vamos comparar a segunda posição com a terceira.
A terceira posição é maior que a segunda? Se for, inverte.
Agora fica: 10 - 30 - 20

Pronto, colocamos em ordem decrescente as duas últimas posições.
Agora vamos comparar a primeira posição com a segunda.

A segunda posição tem um valor maior (30) que a primeira (10)?
Sim! Tem! Então inverte essas duas.
Agora fica: 30 - 10 - 20

Note que ao fazer isso, da direita pra esquerda, pegando duas a duas a posição, jogamos sempre o maior número pro começo da ordem.

Por fim, vamos comparar novamente a segunda com a terceira posição.
A terceira é maior que a segunda? Se for, troca! E é.
Agora fica: 30 - 20 - 10

Algoritmo:
1. Compara as duas últimas posições
2. Compara as duas primeiras posições
3. Compara novamente as duas últimas posições

```
primeiro = int(input('Primeiro numero: '))
    segundo  = int(input('Segundo numero : '))
    terceiro = int(input('Terceiro numero: '))

    print(primeiro,'-',segundo,'-',terceiro)

    if(terceiro > segundo):
        aux = terceiro
        terceiro = segundo
        segundo = aux

    if(segundo > primeiro):
        aux = segundo
        segundo = primeiro
        primeiro = aux

    if(terceiro > segundo):
        aux = terceiro
        terceiro = segundo
        segundo = aux

    print(primeiro,'-',segundo,'-',terceiro)
```

Mais a frente, estudaremos o algoritmo de **bubble sort**, onde vamos aprender a ordenar listas de números de qualquer tamanho. A lógica é a mesma: ir comparando dois a dois, do fim pro começo (ou começo pro fim - depende se quer ordem crescente ou decrescente).

8.

Vamos armazenar o salário na variável 'salario' (que original, hein?)
Teremos outras variáveis no decorrer do programa:
•percentual - percentual de aumento aplicado

•aumento - valor em R$ do aumento aplicado

•salario_novo - novo valor do salário, reajustado pelo aumento

A única coisa que muda, de acordo com o salário fornecido pelo usuário, é o percentual do aumento.

Vamos fazer uma série de testes IF ELIF ELSE pra descobrir qual percentual correto aplicar.

Se for menor ou igual a R$ 280,00 , percentual será 20%
Se não for, vamos testar para ver se é menor ou igual a R$ 700,00 - se for, o percentual será de 15%.

Se também não for menor R$ 700,00, vamos testar para ver se é menor igual a R$ 1500,00 - se sim, 'percentual' recebe 10 (%).

Se também não for menor que R$ 1500,00 é porque é maior então aplicamos 5%.

Para saber o valor do aumento em R$, primeiro pegamos a variável 'percentual' e dividimos por 100.0 (usamos decimal pra mostrar ao Python que essa variável deve ser tratada como um float):
percentual = percentual / 100.0

(essa linha quer dizer: o novo valor de 'percentual' é o valor antigo dela dividido por 100).

Agora multiplicamos 'percentual' por 'salario' e temos o aumento do salário em R$.

Basta somar esse valor ao salário original, que temos o novo salário:

```
salario = float(input('Salário do colaborador: '))
```

```
if (salario <= 280):
    percentual = 20
elif (salario <= 700):
    percentual = 15
elif (salario <= 1500):
    percentual = 10
else:
    percentual = 5

print('Salario original: R$ ', salario)
print('Percentual: ',percentual,'%')

percentual = percentual/100.0
aumento = percentual * salario
novo_salario = salario + aumento

print('Aumento: R$ ',aumento)
print('Novo salário: R$ ', novo_salario)
```

12.

Vamos receber os três lados do triângulo e armazenar nas variáveis **a**, **b** e **c.**

O primeiro teste que fazemos é para saber se a soma de quaisquer dois lados é menor que o terceiro lado. Se for, esses três valores não formam um triângulo e acabou o programa aí, precisa nem testar se é equilátero, isósceles e escaleno.

Se a soma de dois lados quaisquer for maior que o terceiro lado, ok, é triângulo e vamos pro elif.

Agora, vamos testar se é equilátero, para isso comparamos o lado **a** com o **b** e depois o lado **a** com o lado **c**.

Note que não precisamos comparar os lados **b** e **c**, pois se **a** é igual a **b** E TAMBÉM (operador lógico **and**) **a** é igual a **c**, então o lado **b** vai ser automaticamente igual ao lado **c**
.
Se forem todos iguais, diz que é equilátero e acabou aí.

Se não for equilátero, cai no próximo elif.
Lá vamos testar se ele tem dois lados iguais: a==b ou a==c ou b==c (notem o OU, que é o operador lógico **or**).
Se alguma dessas comparações retornar verdadeiro, o triângulo é isósceles

e acaba aí.

Porém, se não for isósceles, cai no else final.
Pois se não é equilátero nem isósceles, e é um triângulo, tem de ser escaleno.

```
a = float(input('Primeiro lado: '))
b = float(input('Segundo  lado: '))
c = float(input('Terceiro lado: '))

# Testando se é triângulo
if (a + b < c) or (a + c < b) or (b + c < a):
    print('Nao é um triangulo')
elif (a == b) and (a == c) :
    print('Equilatero')
elif (a==b) or (a==c) or (b==c):
    print('Isósceles')
else:
    print('Escaleno')
```

13.

Antes de mais nada, vamos relembrar a fórmula de Bháskara para achar as raízes de uma equação do segundo grau, do tipo: **ax² + bx + c = 0** :

$$x = \frac{-b \pm \sqrt{b^2 - 4.a.c}}{2.a}$$

O primeiro teste que fazemos é em relação ao coeficiente **a**. Se for 0, não é uma equação do segundo grau e acaba o programa.

Se for diferente de 0, cai no **else**, que é onde todo nosso programa vai funcionar. Primeiro, dentro do else, pedimos o valor dos coeficientes **b** e **c.**

Agora, vamos calcular o delta.
Em Python, fica assim: delta = b*b **-** (4*a*c)

Agora vamos testar o delta, dentro de um **if** aninhado no **eles** anterior.
Se for menor que 0, encerramos o programa dizendo que as raízes são imaginárias.

Em seguida, usamos um **elif** para testar se delta for 0, se sim valor da raiz será:raiz = **-**b / (2*a)

Por fim, se não é menor que 0 e o delta não é 0, é porque vai ser sempre maior que 0. Essa condição cai no **eles** aninhado, onde calculamos as raízes

assim:
raiz1 = (-b + math.sqrt(delta)) / (2*a)
raiz2 = (-b - math.sqrt(delta)) / (2*a)

Nosso código ficou:

```
import math

print('Equaçao do 2o grau da forma: ax² + bx + c')

a = int( input('Coeficiente a: ') )

if(a==0):
    print('Se a=0, não é equação do segundo grau. Tchau')
else:
    b = int( input('Coeficiente b: ') )
    c = int( input('Coeficiente c: ') )
    delta = b*b - (4*a*c)

    if delta<0:
        print('Delta menor que 0. Raízes imaginárias. Tchau')
    elif delta==0:
        raiz = -b / (2*a)
        print('Delta=0 , raiz = ',raiz)
    else:
        raiz1 = (-b + math.sqrt(delta) ) / (2*a)
        raiz2 = (-b - math.sqrt(delta) ) / (2*a)
        print('Raizes: ',raiz1,' e ',raiz2)
```

14.
Anos bissextos são aqueles que são múltiplos de 4, como 1996, 2000, 2004 etc (que podem ser divididos por 4 deixando resto 0).
Porém, há uma exceção: múltiplos de 100 que não sejam múltiplos de 400.

Uma das duas condições a seguir deve ser verdadeira:
Condição 1: Ser múltiplo de 4 e não ser múltiplo de 100
Condição 2: Ser múltiplo de 400 (se for múltiplo de 400 automaticamente é de 4)

Logo, temos o código:

```
ano = int(input('Ano: '))
if (ano%4==0 and ano%100!=0) or (ano%400==0):
    print('Bissexto')
else:
    print('Não é bissexto')
```

15.
Vamos armazenar os dados nas variáveis 'dia', 'mes' e 'ano'.

Para armazenar o valor lógico verdadeiro ou falso, vamos usar a variável booleana 'valido'. Inicialmente fazer ela ser falsa:
valido = False

O grande segredo nesse algoritmo é o mês.

Primeiro vamos testar se o mês digitado tem 31 dias.
São os meses 1, 3, 5, 7, 8, 10 ou mês 12.
Se tiver digitado um desses valores para 'mes', vamos verificar a variável 'dia' é menor ou igual a 31. Se for, a data é válida e fazemos 'valida = True'
Se não for, continua sendo False

Agora vamos testar os meses que tem 30 dias.
Eles são os meses 4, 6, 9 e o mês 11.
Nesses meses, temos que avaliar se a variável 'dia' tem um número menor ou igual a 30. Se sim, fazemos 'valida = True'.

Por fim, vamos avaliar o mês mais problemático, o mês 2, fevereiro.
Inicialmente, é preciso verificar se é ano bissexto, se for bissexto a variável 'dia' deve ser testada para saber se o valor digitado é 29 ou menos. Se sim, validamos a data com 'valida = True'

Se não for ano bissexto, testamos a variável 'dia' para saber se o valor digitado foi 28 ou menos. Se for, 'valida = True'

Caso não tenham digitado um número de 1 até 12 em mês, a variável 'valida' continua tendo valor False, pois não caiu em nenhum IF ou ELIF.

Por fim, testamos a variável booleana 'valida'. Se for True, dizemos que a data é válida, se tiver o valor lógico False nela, dizemos que é inválida:

```
    dia = int( input('Dia: ') )
    mes = int( input('Mês: ') )
    ano = int( input('Ano: ') )

    valida = False

    # Meses com 31 dias
    if( mes==1 or mes==3 or mes==5 or mes==7 or \
        mes==8 or mes==10 or mes==12):
        if(dia<=31):
            valida = True
```

```
    # Meses com 30 dias
    elif( mes==4 or mes==6 or mes==9 or mes==11):
        if(dia<=30):
            valida = True
    elif mes==2:
        # Testa se é bissexto
        if (ano%4==0 and ano%400!=0) or (ano%400==0):
            if(dia<=29):
                valida = True
        elif(dia<=28):
                valida = True

    if(valida):
        print('Data válida')
    else:
        print('Inválida')
```

16.
Antes de mostrar o exercício, resolver e comentar o código de como descobrir se um número é par ou ímpar em Python, precisamos revistar um tutorial antigo:

Operações matemáticas em Python

Lá estudamos o operador **%** que é o operador de resto da divisão.
Talvez você não lembre direito que troço é isso de resto da divisão.

Mas vamos voltar pra época da escolinha, quando fazíamos continhas de dividir:

O segredo está ali, no resto. O que sobra.
Ele é o segredo de tudo.

Para saber se um número é par ou ímpar, basta dividir ele por 2.
Se for par, o resto é sempre 0, não sobra nada.
Já se for ímpar, vai sempre ter resto 1.

Saber se é Par ou Ímpar em Python

" Faça um Programa que peça um número inteiro e determine se ele é par ou ímpar. Dica: utilize o operador módulo (resto da divisão): %"

Inicialmente pedimos um número ao usuário, e armazenamos na variável 'numero'. Agora, com um simples teste condicional **IF** vamos verificar o resto da divisão dele por 2.

Se o resultado for 0, é par e cai no print do **IF**, dizendo que é par.
Se não for 0 o resto, é porque vai ser 1 e cai no **ELSE** onde printamos que é ímpar:

```
numero = int(input('Digite um inteiro: '))

if (numero%2) == 0:
    print("Par")
else:
    print("Ímpar")
```

Aperfeiçoando código Python

Já dissemos, em algum tutorial anterior, que 1 equivale ao **True** e 0 ao **False**. Quando fazemos o resto da divisão por 2, o resultado é sempre 0 ou 1.

Então nosso código poderia ser assim:

```
numero = int(input('Digite um inteiro: '))

if numero%2 :
    print("Ímpar")
else:
    print("Par")
```

Veja que se o resultado da operação for 1, cai no IF que diz que é ímpar,se for 0 vai pro ELSE que diz que é par.

Bem mais chique, não acha?

Resto da divisão por 3:

"Faça um programa que recebe um inteiro do usuário e diz se esse número é múltiplo de 3 ou não".

Veja o código e tente entender:

```
numero = int(input('Digite um inteiro: '))

if not (numero%3) :
    print("É múltiplo de 3")
else:
    print("Não é múltiplo de 3")
```

Um número é múltiplo de 3 se o resto da divisão dele por 3 for 0.
Então fazemos o teste: numero % 3

Se este resultado for 0, o IF não é executado, pois dá falso.
Então o que fazemos? Invertemos com o operador **not**

Sempre que algo for TRUE o not transforma em FALSE, e vice-versa.
Assim colocamos um not antes do (numero%3) e quando for múltiplo de 3, a expressão **not (numero%3)** vira TRUE e cai no IF dizendo que é múltiplo de 3.

Fodástico, não ?

Curiosidade sobre resto da divisão %

O resto da divisão de um número por 2 pode ser:
0 ou 1

O resto da divisão de um número por 3 pode ser:
0, 1 ou 2

O resto da divisão de um número por 4 pode ser:
0, 1, 2 ou 3

O resto da divisão de um número por 5 pode ser:
0, 1, 2, 3 ou 4
...
...
...

O resto da divisão de um número por n pode ser:
0, 1, 2, 3, ..., (n-1)

17.

Para sabermos o valor da unidade de um número inteiro positivo qualquer, em Python, basta usar o operador de resto da divisão **%** da seguinte maneira:

unidade = numero % 10

Pronto. Só isso.
Se nosso número é 123 e fizermos 123 % 10, o resultado vai ser 3.
Então 3 é nossa unidade.

E como achar a dezena?

O próximo passo é transformar nosso número de 123 pra 12 (excluímos a unidade).
Primeiro, subtraímos do número o valor da unidade.
123 - 3 = 120

Depois, dividimos o número por 10:
120 / 10 = 12

Prontinho, pra achar a dezena basta fazer: 12 % 10 = 2

Em programação Python, fica assim:
numero = (numero - unidade) / 10 # Passa de 123 pra 123-3 = 120 / 10 = 12
dezena = numero % 10 # dezena = 12 % 10 = 2

E pra achar o valor da centena?
De novo, transformamos de 12 pra 1 (excluímos 2).

Subtraímos a dezena achada: 12 - 2= 10
E novamente dividimos por 10: 10/10 = 1

Em programação Python:
numero = (numero - dezena)/10
centena = numero

No caso, paramos por aí. Por só queremos achar até a centena.
Nesse ponto, nosso número vai ser o mesmo valor da centena.

Se quiser para números maiores, só seguir essa lógica.

Código comentado em Python

```
numero = int(input('Digite um numero inteiro positivo: '))

# Extraindo a unidade
unidade = numero % 10

# Eliminando a unidade de nosso número
numero = (numero - unidade)/10

# Extraindo a dezena
dezena = numero % 10

# Eliminando a dezena do número original, fica a centena
numero = (numero - dezena)/10
centena = numero

# Fazendo ser inteiros
dezena = int(dezena)
centena = int(centena)
print(centena,'centena(s),',dezena,'dezena(s) e',unidade,'unidade(s)')
```

18.

Há várias maneiras diferentes de darmos 256 reais, ou qualquer outro valor. Poderíamos, por exemplo dar 256 cédulas de 1 real (hoje em dia só moeda, mas antes tinha cédula de 1 real sim).

Mas seria altamente inconveniente sair de um caixa eletrônico com centenas de cédulas. Ou seja, queremos usar o menor número de cédulas possíveis!

Isso é feito da seguinte maneira: dando cédulas de valor alto.

•Notas de R$ 100,00

O primeiro passo é tentar empurrar o maior número de notas de 100 reais possível. No caso de 256 reais, só podemos dar 2 notas de 100 reais, pois se dermos mais, passa de 256.

Como fazemos isso?
Dividindo o valor que o usuário pediu pra sacar por 100.

Vamos colocar em 'numero', o valor que ele solicitou que fosse sacado.

Se fizermos:
cem = numero / 100

Teremos que cem=2.56
Não seria legal se a gente tirasse esse ponto fluante e essa parte decimal?
É fácil fazer isso, só usar a função **int()**, que vai transformar esse número em inteiro (ele pega a parte inteira e descarta o resto).

Assim, o número de cédulas de 100 reais é:
cem = int(numero/10)

Se antes queríamos sacar 256 reais, agora queremos somente 56, pois já demos os 200 reais.

Vamos transformar nosso valor de 256 pra 56 da seguinte maneira:
numero = numero - (cem*100)
(numero = 256 - 2*100 = 56)

Concorda?

•Notas de R$ 50,00

Agora vamos tentar usar o máximo possível de notas de 50 reais.
Para isso, basta dividir nosso número por 50 e pegar a parte inteira, como fizemos com as cédulas de 100:

cinquenta =int(numero/50)

No nosso exemplo de 256 reais: cinquenta = int(56/50) = int(1.12) = 1

Agora que já usamos as de 50 reais, vamos tirar ela de nosso número:
numero = numero - (cinquenta*50)
(numero = 56 – 1.50)

•Notas de R$ 10,00

A lógica é a mesma:

dez = int(numero/10)
numero = numero - (dez*10)

•Notas de R$ 5,00

cinco = int(numero/5)
numero = numero - (cinco*5)

E o que sobrou, é nota de um real:
um = numero

Nosso código Python fica:

```
numero = int(input('Valor para sacar [10-600]: '))

cem = int(numero / 100)
numero = numero - (cem*100)

cinquenta = int(numero/50)
numero = numero - (cinquenta*50)

dez = int(numero/10)
numero = numero - (dez*10)

cinco = int(numero/5)
numero = numero - (cinco*5)

um = numero

print('Notas R$100,00 = ',cem)
print('Notas R$ 50,00 = ',cinquenta)
print('Notas R$ 10,00 = ',dez)
print('Notas R$  5,00 = ',cinco)
print('Notas R$  1,00 = ',um)
```

Nosso código também poderia ser:

```
numero = int(input('Valor para sacar [10-600]: '))

cem = int(numero / 100)
numero = numero % 100
```

```
cinquenta = int(numero/50)
numero = numero % 50

dez = int(numero/10)
numero = numero % 10

cinco = int(numero/5)
numero = numero % 5

um = numero

print('Notas R$100,00 = ',cem)
print('Notas R$ 50,00 = ',cinquenta)
print('Notas R$ 10,00 = ',dez)
print('Notas R$  5,00 = ',cinco)
print('Notas R$  1,00 = ',um)
```

Consegue entender o motivo?

19. Arredondar números em Python

A utilidade mais básica da função **round()** é receber um número qualquer e arredondar ele. O mais lógico é arredondar um flutuante para um inteiro.

Basta colocar um número dentro dela: round(numero) que ela devolve ele arredondado.
Vamos ver alguns exemplos:

```
Numero original: 1.9
Arredondado    : 2

Numero original: 21.12
Arredondado    : 21

Numero original: 3.4
Arredondado    : 3

Numero original: 4.6
Arredondado    : 5
```

O código para você testar a round() é:

```
numero = float(input('Numero original: '))
print("Arredondado    :", round(numero))
```

Mas, por exemplo, o número 1.5 ?
Ele está igual distância de 1 como de 2, qual será o resultado se aplicarmos **round(1.5)** ?

A resposta é 2.
Então você deve pensar: *ah ok, arredonda pra cima.*

Então teste **round(2.5)**, o resultado vai ser 3, pra cima, não é?
Errado, é 2 de novo.

"*Que bruxaria é essa, Python Progressivo? Tá bugada a função round() ?*"
Não!

A função **round()** tem uma característica especial: se o número flutuante estiver igual distância entre o inteiro de cima e o inteiro de baixo, ela arredonda pro **número par** mais próximo!

Veja:

```
Numero original: 1.5
Arredondado    : 2

Numero original: 2.5
Arredondado    : 2

Numero original: -3.5
Arredondado    : -4

Numero original: -5.5
Arredondado    : -6
```

Arredondar com casas decimais em Python

Colocando apenas um número dentro da round() ela arredondou e nos devolveu um inteiro.

Mas, e se quisermos arredondar pra ter um valor float?
Por exemplo, quero arredondar 1.9999 pra 2.0 ?
Ou 21.129 pra 21.13 ?

A round() faz isso!
Seu estereótipo é: **round(numero, n)**
Onde 'numero' é o número que você seja arredondar e 'n' é o número de casas decimais **após** o ponto flutuante, que deseja arredondar.

Por exemplo:
numero = 1.23456789

round(numero, 1) = 1.2
round(numero, 2) = 1.23
round(numero, 3) = 1.235
round(numero, 4) = 1.2346
round(numero, 5) = 1.23457
round(numero, 6) = 1.234568
round(numero, 7) = 1.2345679
round(numero, 8) = 1.23456789

Precisão cirúrgica, esse Python, não ?

Arredondar pra baixo e pra cima

Muitas vezes queremos sempre arredondar pra baixo, por exemplo:
1.2 -> 1
1.1 -> 1
9.9 -> 9

Outras vezes podemos querer sempre arredondar pra cima:
1.1 -> 2
1.2 -> 2
9.9 -> 10

Pra arredondar pra baixo um número **num** basta fazer:
round(num - 0.5)

Para arredondar pra cima:
round(num + 0.5)

Mas só funciona se for para números decimais, ok ?

Exercício Resolvido de Python

Escreva um script que peça um número ao usuário.
Em seguida, ele deve descobrir se o número é inteiro ou decimal.

Se for decimal, deve dizer o número arredondado pra cima e arredondado pra baixo.

Primeiro, vamos descobrir se é inteiro ou decimal. Um número **num** vai ser inteiro quando ele for igual ao seu número arredondado: num == round(num) retorna **True** para um inteiro e **False** para um decimal.

Caso seja decimal, usamos round(num-0.5) pra arredondar pra baixo e round(num+0.5) pra arredondar pra cima**.**

Nosso código Python fica:

```
num = float(input('Numero original: '))

if num == round(num):
    print("Inteiro")
else:
    print("Decimal")
    print("Arredondado pra baixo: ", round(num-0.5) )
    print("Arredondado pra cima : ", round(num+0.5) )
```