

LABORATÓRIO DE PROGRAMAÇÃO

Revisão de Python

Aula 1

Prof^a. Kamilla Dória



É melhor você tentar algo, vê-lo não funcionar e aprender com isso, do que não fazer nada."

—MARK ZUCKRBERG



APRESENTAÇÕES

APRESENTAÇÃO

Kamilla Dória da Silveira

Formação:

- Formada em Sistemas de TI (Faculdade de Negócios de Sergipe)
- Mestre em Ciência da Computação (UFPE)

Contato:

kamilla.silveira@souunit.com.br

Experiência:

- Experiência em análise de sistemas, desenvolvimento, liderança de equipes e gerenciamento de projetos.
- Atualmente trabalho como Líder de Desenvolvimento SAP na Cencosud Brasil, gerenciando equipe de projetos de sistemas.

APRESENTE-SE

“Qual seu nome e qual sua idade?”

“Conhecimento básico ou avançado em Python?”



SOBRE A DISCIPLINA

1

Propiciar aos discentes a oportunidade de utilizar, em aplicações reais, técnicas de implementação de software

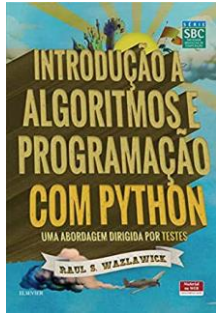
2

Estimular a qualidade de produto de software e a pesquisa em temas relacionados à engenharia de software por meio de projetos práticos

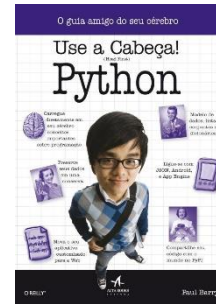
EMENTA

- Revisão de estruturas básicas no Python
- Código limpo
- Funções
- Listas
- Tuplas
- Dicionários
- Matriz
- Arquivos
- Estrutura de dados externas
- Persistência de dados
- Exceção
- Programação orientada a objetos
- Ordenação
- Busca

REFERÊNCIAS BIBLIOGRÁFICAS



WAZLAWIK, Raul S. Introdução a Algoritmos e Programação com Python. Elsevier Editora Ltda., 2017, 232 p.



BARRY, Paul. Use a cabeça!: Python. Rio de Janeiro: Alta Books, 2012.



RAMALHO, Luciano. Python Fluente: Programação clara, concisa e eficaz. São Paulo: Novatec, 2015.

- Revisão de estruturas básicas no Python
- Código limpo
- Funções
- Listas
- Tuplas
- Dicionários
- Matriz
- Arquivos

- Estrutura de dados externas
- Persistência de dados
- Exceção
- Programação orientada a objetos
- Ordenação
- Busca

AVALIAÇÃO

- Formação da nota final:
 - Prova em grupo: 8,0 pontos;
 - Medida de eficiência em grupo: 2,0 pontos.

TÓPICOS

- IDE
- Variáveis
- Estruturas condicionais
- Estruturas de repetição



IDE

RELEMBRANDO A INSTALAÇÃO DA IDE PYTHON

- Instalar o interpretador Python;
- Instalar uma IDE.

INSTALAÇÃO DO INTERPRETADOR

- Baixar e instalar o interpretador Python:

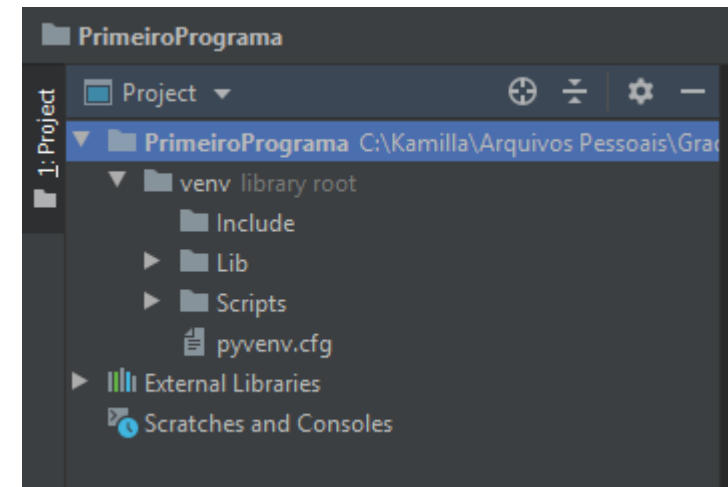
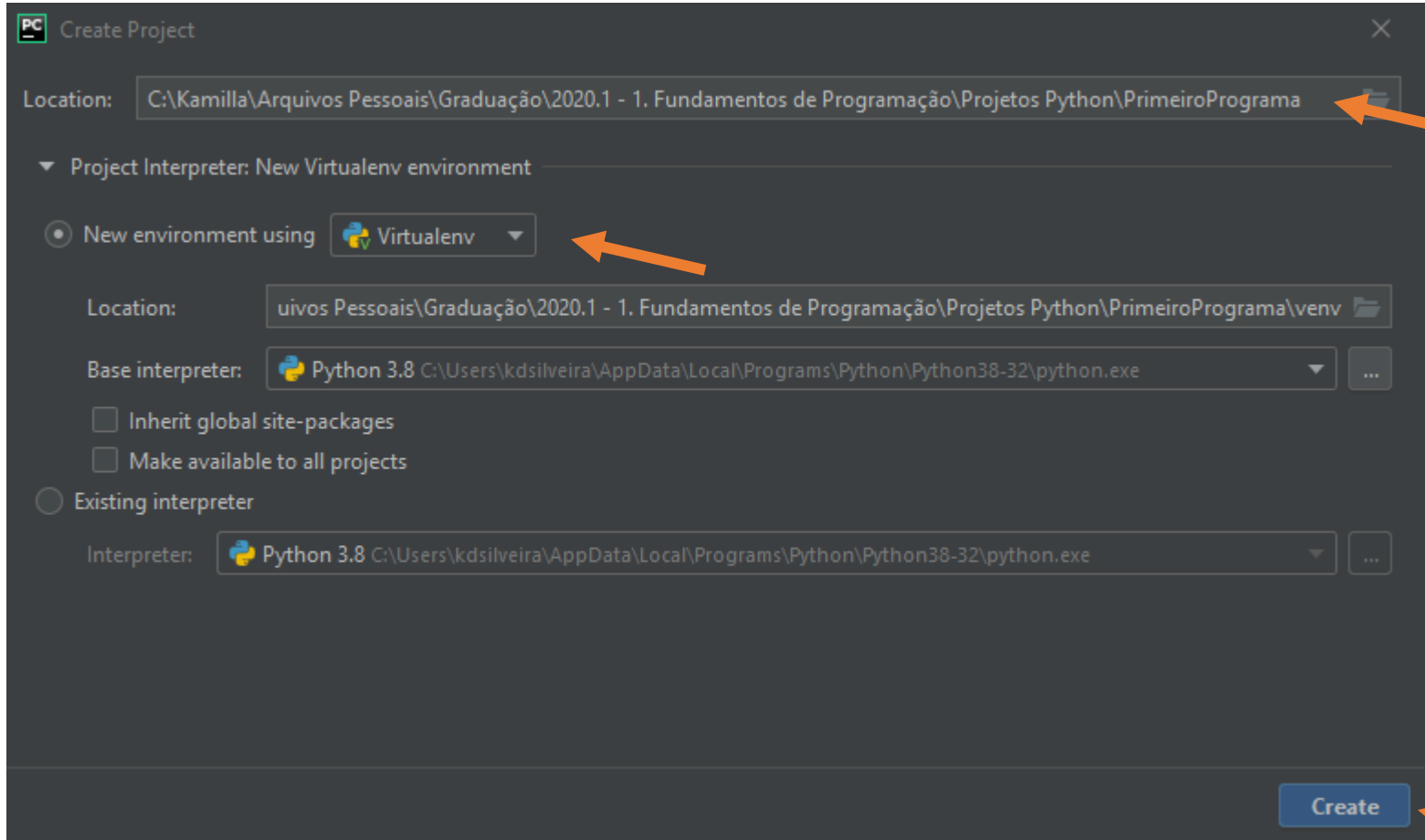
<http://www.python.org>

INSTALAÇÃO DA IDE

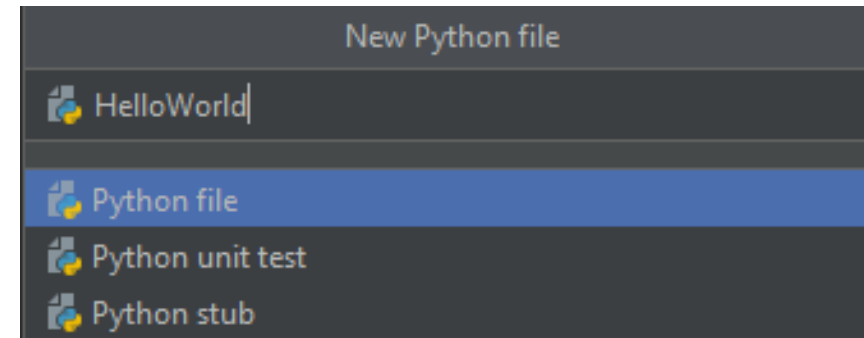
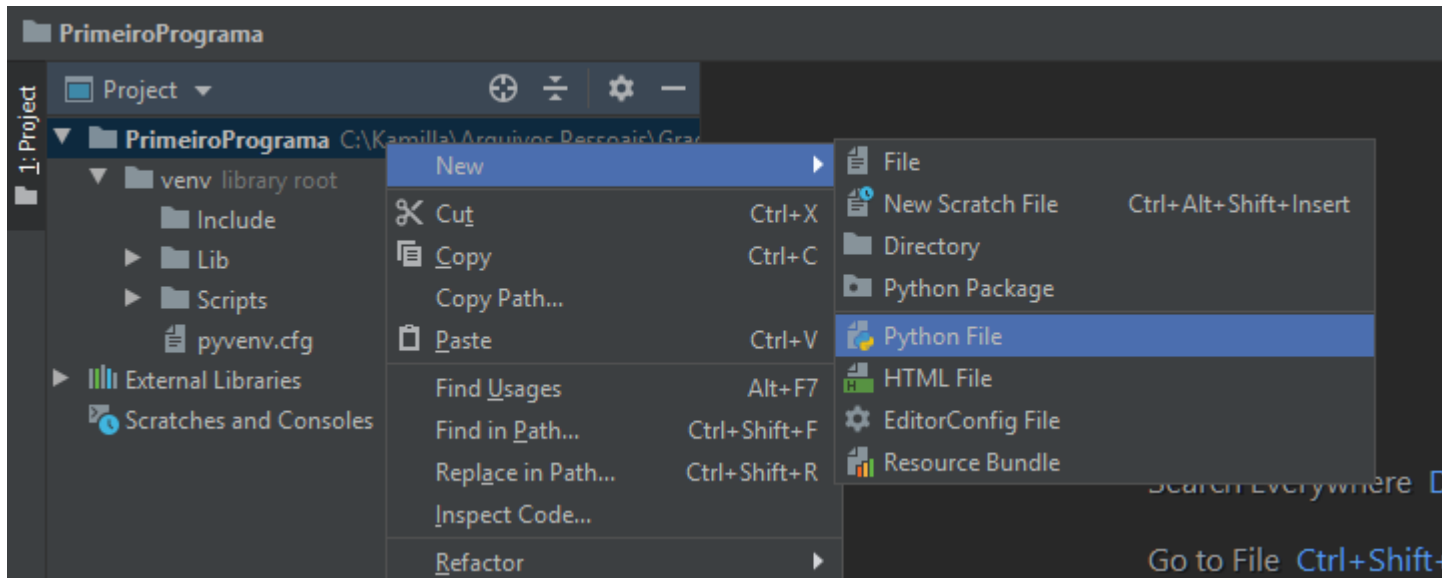
- Baixar e instalar a IDE pyCharm para programação em Python:

<https://www.jetbrains.com/pycharm-edu/download>

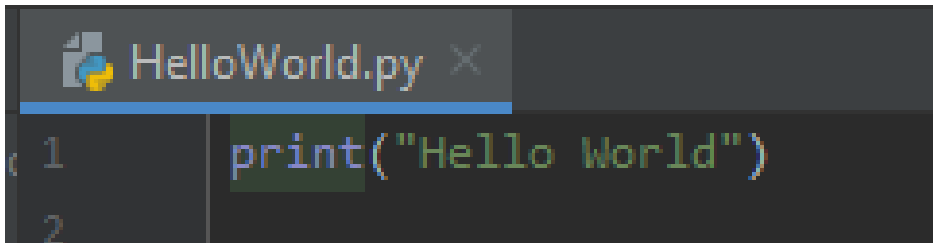
CRIANDO O PROJETO



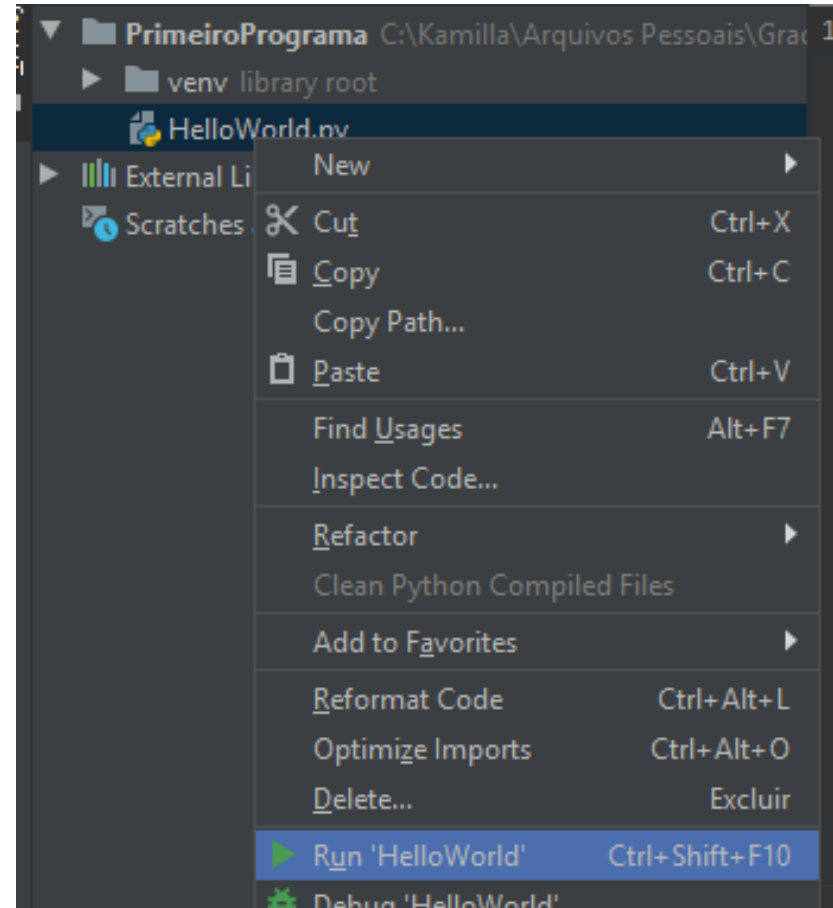
CRIANDO O PROGRAMA



HELLO WORLD



```
1 print("Hello World")
2
```





VARIÁVEIS

VARIÁVEL

- Seu conteúdo pode ser escrito através do comando *print*;
- Em algumas linguagem precisa ser declarada antes de usar;
 - Em Python não há essa necessidade, basta atribuir um valor;
- Pode conter valores diferentes ao longo da execução do programa.

```
x = 4  
print(x)  
x = 6  
print(x)
```

```
"Aula3/Exercicio.py"  
4  
6  
  
Process finished with exit code 0
```

TIPOS DE VALORES NUMÉRICOS

- Inteiros (*int*): 2, -59
- Números de ponto flutuante (*float*): 2.0, -59.33
- Booleanos (*bool*): *true* e *false*
- Números complexos (*complex*): $3+4j$, $-2+4.5j$

OPERAÇÕES ARITMÉTICAS

- Soma: +
- Subtração: -
- Multiplicação: *
- Divisão: /
- Divisão inteira (ignora as casas decimais do resultado): //
- Resto da divisão inteira: %
- Potenciação: **

FORMATAÇÃO DE SAÍDA

- Formatação especial de saída: `.format(x, y)`
 - A mensagem *format* é associada a uma literal, número ou expressão através de um `""`.
 - Os argumentos são inseridos dentro da literal na seguinte formatação: `{0}`
 - Podem ser utilizados vários argumentos enumerados entre chaves.

```
x = 1
y = 2
print('Teste {0} e Teste {1}'.format(x,y))
```

"Aula3/Exercicio.py"

Teste 1 e Teste 2

Process finished with exit code 0

ENTRADA DE DADOS

```
x = input('Digite um número: ')\ny = input('Digite outro número: ')\nprint('Soma: ', x+y)
```

```
"Aula3/Exercicio.py"\nDigite um número: 2\nDigite outro número: 3\nSoma: 23
```

Process finished with exit code 0

```
x = int(input('Digite um número: '))\ny = float(input('Digite outro número: '))\nprint('Soma: ', x+y)
```

```
"Aula3/Exercicio.py"\nDigite um número: 2\nDigite outro número: 3\nSoma: 5.0
```

Process finished with exit code 0

PRECEDÊNCIA DE OPERADORES

- Usa a regra de precedência da matemática:
 - Primeiro são executadas operações de potenciação;
 - Depois as operações de multiplicação e divisão;
 - Por ultimo as adições e subtrações;
 - Expressões em parênteses são executadas antes das que estão fora.

```
print(2 + 6 * 4 / 2)  
print((2 + 6) * 4 / 2)
```

```
"Aula3/Exercicio.py"
```

```
14.0
```

```
16.0
```

```
Process finished with exit code 0
```

CONSTANTES

- Identificadores que após receber um valor não podem mais ser alterados.
- Python não possui constante.
 - Convenção: utilizar letras maiúsculas para identificar e nunca alterar seu valor.
- Exemplo:
 - $\text{PI} = 3.14159$

ARREDONDAMENTO

- Números de ponto flutuante têm sua precisão limitada em cerca de 16 casas decimais.
 - Função do arredondamento: *round()*
 - Final da casa decimal menor ou igual que 5, arredonda para baixo;
 - Final da casa decimal maior que 5, arredonda para cima;
 - Permite arredondar para ponto flutuante com casas decimais delimitadas: *round(x, 2)*.

```
preco = float(input("Informe o preço: "))  
print(f'Desconto de 10%: {round(preco*0.9, 2)}')
```

"Aula3/Exercicio.py"

Informe o preço: 7

Desconto de 10%: 6.3

Process finished with exit code 0



ESTRUTURAS CONDICIONAIS

COMANDOS DE SELEÇÃO

- Comando: *if(condição):*
 <bloco de código>
 else:
 <bloco de código>
- O else não é obrigatório.
- A condição deve resultar em uma expressão logica booleana.

OPERADORES DE COMPARAÇÃO

- Têm precedência mais baixa do que os operadores aritméticos.
 - Igual: ==
 - Diferente: !=
 - Maior: >
 - Menor: <
 - Maior ou igual: >=
 - Menor ou igual: <=

OPERADORES DE COMPARAÇÃO

```
idade = int(input("Digite sua idade: "))  
if (idade >= 18):  
    print('Você é maior de idade')  
else:  
    print('Você é menor de idade')
```

"Aula3/Exercicio.py"

Digite sua idade: 15

Você é menor de idade

Process finished with exit code 0

COMPARAÇÃO COM NÚMEROS PONTO FLUTUANTE

- É preciso ter cuidado com a comparação de números de ponto flutuante em casos de expressões devido as casas decimais.
 - Utilizar o arredondamento na comparação;
 - Calcular o valor absoluto da diferença entre dois números e verificar se é menor que 0,01.
 - A função `abs(expressão)` retorna o resultado da expressão sem sinal (positivo ou negativo).

SELEÇÕES ANINHADAS

- Estruturas de seleção podem ficar umas subordinadas às outras.
- Comando:

```
if condição:
    if condição:
        <bloco de código>
    else:
        <bloco de código>
else:
    <bloco de código>
```

- As instruções internas só serão executadas caso a 1ª condição seja atendida.

CONDIÇÃO COMPOSTA

- Operadores:
 - Conjunção "e": condição1 *and* condição2
 - O resultado será true tanto a condição 1 quanto a condição 2 for verdadeira.
 - Disjunção "ou": condição1 *or* condição2
 - O resultado será true se qualquer uma das condições forem verdadeiras.

```
nota = int(input('Digite a nota: '))  
if nota >= 3 and nota < 7:  
    print('Aluno em recuperação')
```

"Aula3/Exercicio.py"

Digite a nota: 5

Aluno em recuperação

Process finished with exit code 0

CONDIÇÃO COMPOSTA

- Condição composta resumida:

```
nota = int(input('Digite a nota: '))  
if (3 <= nota < 7):  
    print('Aluno em recuperação')
```

```
"Aula3/Exercicio.py"  
Digite a nota: 2
```

```
Process finished with exit code 0
```

CONDIÇÃO COMPOSTA

- Operador:
 - Negação “não”: not (expressão)

```
x = True  
print(not x)
```

```
"Aula3/Exercicio.py"
```

```
False
```

```
Process finished with exit code 0
```

PRECEDÊNCIA DE OPERADORES

- Potenciação
- Multiplicação e Divisão
- Soma e Subtração
- Comparadores
- Negação
- Conjunção
- Disjunção

SELEÇÃO MÚLTIPLA

- Utilizada quando é necessário tomar uma decisão baseada em várias condições;
- Evita ter que usar várias seleções aninhadas.
- Comando:

```
if (condição1):  
    <bloco de código>  
elif (condição2):  
    <bloco de código>
```


SELEÇÃO MÚLTIPLA

```
x = int(input('Primeiro número: '))
y = int(input('Segundo número: '))
operador = input('Operador aritmético: ')
if (operador == '+'):
    print(x+y)
elif (operador == '-'):
    print(x-y)
```

"Aula3/Exercicio.py"

Primeiro número: 2

Segundo número: 3

Operador aritmético: +

5

Process finished with exit code 0



ESTRUTURAS DE REPETIÇÃO

- Comando: for ou para-faça

```
for variável_iteração in campo_iteração:  
    comando_a_ser_repetido
```

- Iterador ou Contador: variável especial utilizada no laço para controlar as repetições;
 - Deve ser nomeada; normalmente "i", "j", "k", etc.
- Campo de Iteração: lista de valores pela qual o comando for irá percorrer utilizando o contador.

CONTAGEM SIMPLES

- Exemplo:
- Repetir um comando 100 vezes:

```
for i in range(100):  
    print(i)
```

- Comando *range* é utilizado para gerar uma lista de números inteiros;
 - *range*(100) : gera uma lista de 100 números (0 a 99);
 - *range*(1, 101) : gera uma lista de 100 números (1 a 100).
 - O Valor do 1º argumento não pode ser maior que o valor do 2º argumento.

CONTAGEM VARIADA

- É possível indicar o passo do incremento no comando for.
- Exemplo: Incrementar utilizando números pares:

```
for i in range(0, 10, 2):  
    print(i)
```

"Aula4/Exercicio.py"

0
2
4
6
8

Process finished with exit code 0

- Uso do comando for para acumular valores de cálculos repetitivos;
- Exemplo: valor de uma compra no supermercado:
 - A variável total_compra deve ser inicializada antes do comando for.

```
total_compra = 0
for preco_item in range(1, 101):
    total_compra += preco_item
print(total_compra)
```

```
"Aula4/Exercicio.py"
```

```
5050
```

```
Process finished with exit code 0
```

FATORIAL

- Consiste no produto de todos os números de 1 ao número dado;

```
n = int(input('Digite um número para imprimir o fatorial: '))
fatorial = 1
for i in range(1, n+1):
    fatorial *= i
print(fatorial)
```

```
"Aula4/Exercicio.py"
Digite um número para imprimir o fatorial: 3
6

Process finished with exit code 0
```

- Substitui a soma do somatório pela multiplicação;
- A variável do fatorial tem que ser inicializada com 1 e não 0.

ANINHAMENTO

- Estruturas de repetição com outras estruturas de repetição subordinadas;
- Exemplo: imprimir as horas, minutos e segundos de um dia (relógio digital):

```
for hora in range(24):  
    for minuto in range(60):  
        for segundo in range(60):  
            print(f'{hora}:{minuto}:{segundo}')
```


EXERCÍCIO 1

Você está iniciando um novo negócio: uma locadora de veículos. Para iniciar o controle do aluguel de carros é necessário um sistema simples. Dessa forma, desenvolva um programa em Python para:

- a) Solicitar ao usuário informações sobre as locações: nome do cliente, sexo (F- Feminino, M - Masculino), placa do carro alugado, quantidade de quilômetros contratados, quantidade de dias contratados;
- b) Calcular e imprimir a placa do carro e valor total a pagar para CADA cliente, considerando que deverá ser cobrado o valor de R\$ 70,00 por dia contratado, e R\$ 0,10 para cada quilômetro contratado;
- c) Calcular e imprimir a média de quilômetros contratados pelos clientes;
- d) Calcular e imprimir o nome das clientes de sexo feminino que fecharam aluguéis acima de 7 dias contratados.

Obs.: o programa encerra quando o usuário informa o texto SAIR.

EXERCÍCIO 2

A LOTOFÁCIL consiste na extração de 15 números aleatórios diferentes, no universo de 01 a 25. Você marca entre 15 a 18 números, dentre os 25 disponíveis no volante, e fatura o prêmio se acertar 11, 12, 13, 14 ou 15 números. Pode ainda deixar que o sistema escolha os números para você por meio da Surpresinha. Considerando estas informações, faça um programa em Python para:

- a) Solicitar ao usuário a quantidade de dezenas que ele deseja marcar na primeira aposta (entre 15 e 18 números). Caso o usuário informe uma quantidade de dezenas fora do intervalo válido, o programa deve solicitar nova digitação, tantas vezes quantas forem necessárias;
- b) Solicitar ao usuário informar os números da primeira aposta (dezenas de 01 a 25, sem repetição). Caso o usuário informe um número repetido, o programa deverá apresentar uma mensagem "Número repetido" e solicitar nova digitação. Assim como se o usuário informar um número fora do intervalo válido, o programa deverá apresentar uma mensagem "Dezena inválida" e solicitar nova digitação.
- c) Gerar aleatoriamente duas apostas, com 18 números, usando a "Surpresinha".
- d) Simular o resultado (15 dezenas sorteadas) de um concurso da Lotofácil;
- e) Imprimir (em ordem crescente) as dezenas da primeira aposta, das duas apostas (surpresinha) e do resultado do concurso da Lotofácil simulado.



OBRIGADA!