

Minicurso: Python

INTRODUÇÃO AO PYTHON E REDES NEURAIS ARTIFICIAIS

Professor Maximilian Jaderson de Melo

Professor Guilherme Figueiredo Terenciani

Conteúdo deste curso

- Introdução python
- Redes Neurais Artificiais
 - Perceptron
 - Exemplo prático.

Sobre o curso

Python é poderoso... e rápido;
Roda em Tudo;
É amigável e Fácil de aprender;
É aberto



Fonte: <https://www.python.org/>

Sobre o curso

O intérprete Python geralmente é instalado no diretório:

/usr/local/bin/python3.7

Comumente para executarmos digitamos:

```
python3
```



Fonte: <https://www.python.org/>

Meu primeiro código

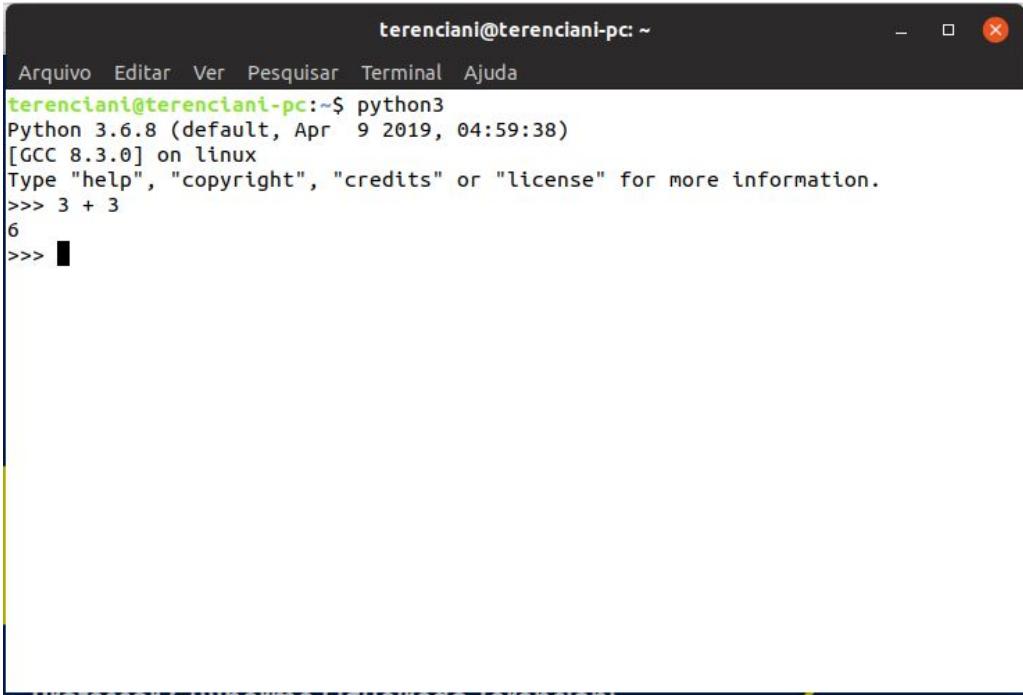
Abra o terminal linux:

ctrl + alt + t

Digite:

python3

Divirta-se!



A screenshot of a terminal window titled "terenciani@terenciani-pc: ~". The window has a dark theme with white text. At the top, there is a menu bar with options: Arquivo, Editar, Ver, Pesquisar, Terminal, and Ajuda. Below the menu, the command "python3" is entered at the prompt. The terminal displays the Python 3.6.8 version information, which includes the date (April 9, 2019) and time (04:59:38). It also shows the compiler used ([GCC 8.3.0] on linux) and a message encouraging users to type "help", "copyright", "credits" or "license" for more information. A simple calculation, "3 + 3", is run, resulting in "6". The terminal ends with a prompt ">>>".

```
terenciani@terenciani-pc:~$ python3
Python 3.6.8 (default, Apr  9 2019, 04:59:38)
[GCC 8.3.0] on linux
Type "help", "copyright", "credits" or "license" for more information.
>>> 3 + 3
6
>>> 
```

Salvando em um arquivo

Abra o terminal linux:

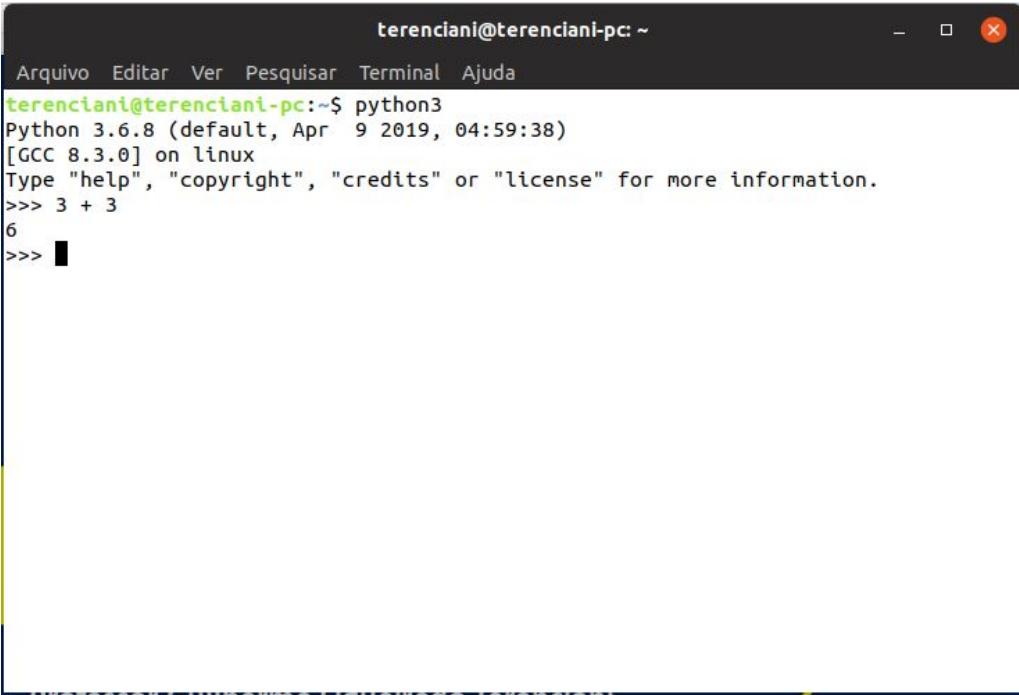
```
mkdir cursopython  
ls  
cd cursopython
```

Digite:

```
code meucodigo.py
```

Para rodar:

```
python3 meucodigo.py
```



A screenshot of a Linux terminal window titled "terenciani@terenciani-pc: ~". The window has a dark theme with white text. At the top, there is a menu bar with options: Arquivo, Editar, Ver, Pesquisar, Terminal, and Ajuda. Below the menu, the command "python3" is entered, followed by its version information: "Python 3.6.8 (default, Apr 9 2019, 04:59:38) [GCC 8.3.0] on linux". It also says "Type "help", "copyright", "credits" or "license" for more information.". Then, the command "3 + 3" is entered, resulting in the output "6". The terminal ends with a prompt ">>> █".

Tipos python

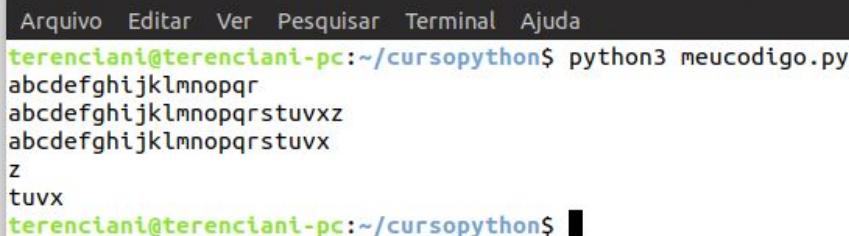
- **Variáveis:** Parecido com o PHP mas com tipagem forte;
- **Números:** O interpretador python funciona basicamente igual uma calculadora. Os sinais: “* / - +” serão igual as linguagens;
 - Divisão inteira //;
 - Exponenciação **;
 - Resto %.
- **Strings:** entre aspas duplas ou simples;
 - Strings é parecida com um vetor;
 - Podemos criar substrings utilizando slicing;
 - print(x[0 : 10]);

Impressão e slicing

- O comando para imprimir dados na saída padrão do Python é o **print**;
- Podemos mostrar qualquer valor utilizando o print;
- O slicing cria substrings de uma string maior;

meucodigo.py > ...

```
1 x = "abcdefghijklmnopqrstuvwxyz";
2 print(x[0:18]);
3 print(x[:])
4 print(x[:-1]);
5 print(x[-1]);
6 print(x[-5:-1]);
7
8
```



A screenshot of a terminal window. The title bar says "Arquivo Editar Ver Pesquisar Terminal Ajuda". The command "terenciani@terenciani-pc:~/cursopython\$ python3 meucodigo.py" is entered. The output shows the string "abcdefghijklmnopqrstuvwxyz" being printed in three different ways: with slicing, with a colon, and with a step of -1. The terminal prompt "terenciani@terenciani-pc:~/cursopython\$" is visible at the bottom.

```
Arquivo Editar Ver Pesquisar Terminal Ajuda
terenciani@terenciani-pc:~/cursopython$ python3 meucodigo.py
abcdefghijklmnopqrstuvwxyz
abcdefghijklmnopqrstuvwxyz
abcdefghijklmnopqrstuvwxyz
z
tuvx
terenciani@terenciani-pc:~/cursopython$
```

Impressão e slicing

- Não precisamos utilizar o



Listas

- **Listas:** Muito parecido com o slicing das strings;

- Declaração `lista = [1,2,3,4,5,6];`
- Concatena `lista + [1,2];`
- Inserir no fim `lista.append(3);`
- tamanho da lista `len(lista);`

Laços de repetição

```
meucodigo.py •  
meucodigo.py > ...  
1 if condition:  
2     pass  
3 else:  
4     pass  
5  
6 while expression:  
7     pass  
8 for target_list in expression_list:  
9     pass  
10  
11 if expression:  
12     pass
```

Laços de repetição

```
meucodigo.py > ...
meucodigo.py > ...
1 print("Exemplos de if")
2 x = 19;
3 if x<20:
4     print("Sou maior que 20")
5 else:
6     print("Sou menor que 20")
7 print("While:")
8 i=15;
9 while (i<x):
10    print(i);
11    #i++ Exemplo Comentário
12    # ++ não existe em Python
13    i = i+1
14 print("For em uma lista")
15 for x in [1,2,3]:
16    print(x)
```

```
terenciani@terenciani-pc: ~/cursopython
Arquivo Editar Ver Pesquisar Terminal Ajuda
terenciani@terenciani-pc:~/cursopython$ python3 meucodigo.py
Exemplos de if
Sou maior que 20
While:
15
16
17
18
For em uma lista
1
2
3
terenciani@terenciani-pc:~/cursopython$
```

Entrada e saída de dados

- Entrada:

- **input**

- dadodeentrada = input();

- Cast

- int(dadodeentrada);
 - str(dadodeentrada);
 - float(dadodeentrada);

- Saída

- **print** print(dadodeentrada);

Atividade

Dez minutos de exercícios:

1. Leia dois números calcule a média imprimindo se o aluno tirou uma nota maior que 7 (Aprovado) ou (Reprovado)

Lembrando:

Como Executar:

Terminal Ajuda

cursopython\$ code exercicio1.py
cursopython\$ python3 exercicio1.py

2. Leia um número inteiro e imprima os valores de zero até o mesmo;

3. Desafio:

Leia um número e calcule se o mesmo é primo ou não.

4. Desafio desafiador:

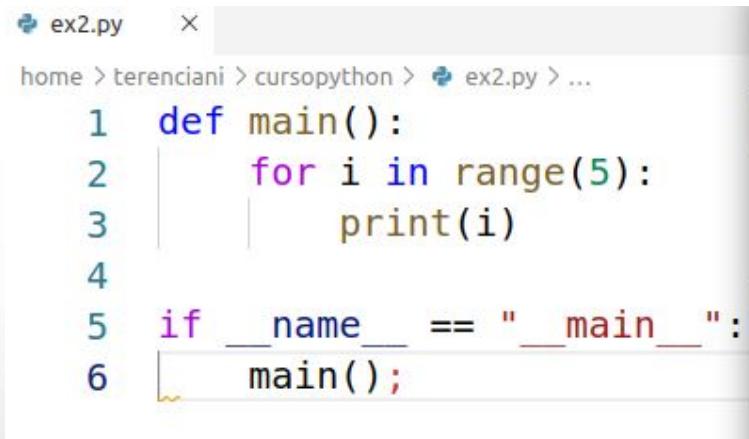
Prove que $p=np$; para qualquer n ;



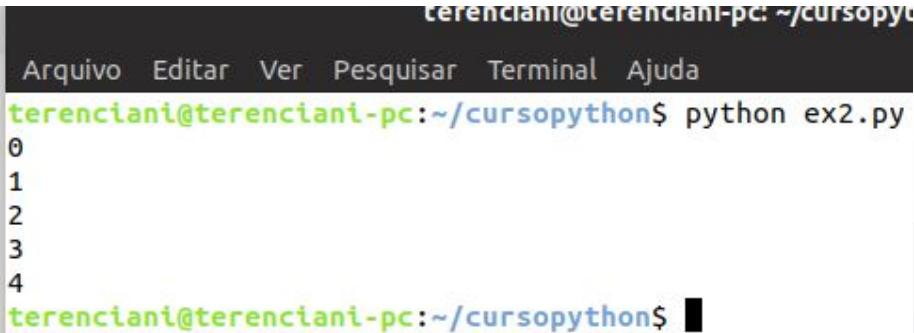
Listas

- **Range**

- Feito para iterações entre sequências de números;



```
ex2.py  x
home > terenciani > cursopython > ex2.py > ...
1 def main():
2     for i in range(5):
3         print(i)
4
5 if __name__ == "__main__":
6     main()
```

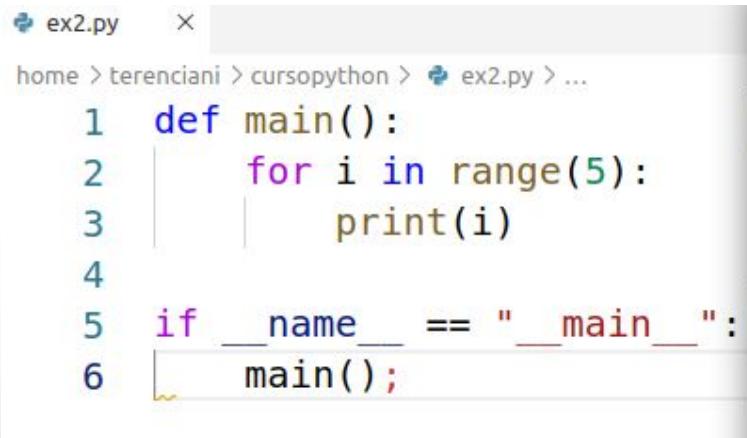


```
terenciani@terenciani-pc:~/cursopython$ python ex2.py
0
1
2
3
4
```

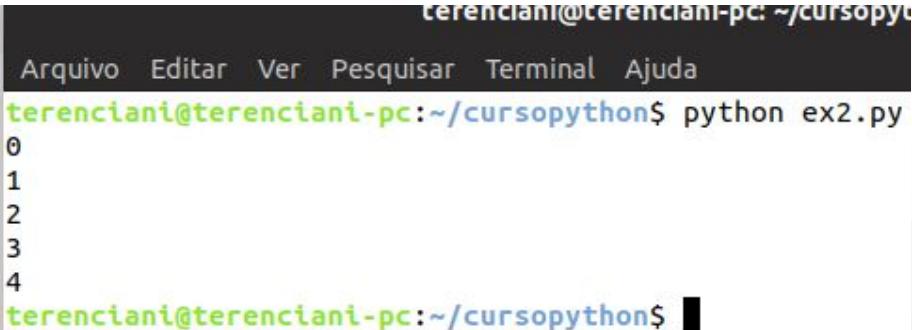
Listas

- **Funções**

- Podemos definir funções utilizando a palavra reservada **def**



```
ex2.py    x
home > terenciani > cursopython > ex2.py > ...
1 def main():
2     for i in range(5):
3         print(i)
4
5 if __name__ == "__main__":
6     main()
```



```
terenciani@terenciani-pc:~/cursopython$ python ex2.py
0
1
2
3
4
```

Parâmetros

Passagem de argumentos para funções:

```
2.py  x  
! > terenciani > cursopython > ex2.py > ...  
1 def printtere(inteiro):  
2     print(inteiro);  
3  
4 def main():  
5     for i in range(5):  
6         printtere(i)  
7  
8 if __name__ == "__main__":  
9     main();
```

```
terenciani@terenciani-pc: ~/cursopython  
Arquivo Editar Ver Pesquisar Terminal Ajuda  
terenciani@terenciani-pc:~/cursopython$ python ex2.py  
0  
1  
2  
3  
4  
terenciani@terenciani-pc:~/cursopython$ █
```

Listas



1. Leia N e calcule o somatório de 1 até N utilizando uma função.
2. Exercício desafiador: Utilize funções para realizar o cálculo de fibonacci

Segundo a Wikipédia, a Sequência de Fibonacci é "uma sequência de números inteiros, começando normalmente por 0 e 1, na qual, cada termo subsequente corresponde a soma dos dois anteriores. A sequência recebeu o nome do matemático italiano Leonardo de Pisa, mais conhecido por Fibonacci, que descreveu, no ano de 1202, o crescimento de uma população de coelhos, a partir desta. Tal sequência já era no entanto, conhecida na antiguidade."

Portanto, começando por 1, os números que compõem a sequência de Fibonacci são:

1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89, 144, 233, 377, 610, 987, 1597, 2584, ...

A sequência é definida recursivamente pela fórmula abaixo:

$$F_n = F_{n-1} + F_{n-2}$$

$$F_1 = 1, F_2 = 1$$

A sequência de Fibonacci ocorre com bastante frequência na natureza, no padrão de formação de flores, caracóis...

Crie um algoritmo em python que a partir da leitura de um valor N imprima os N primeiros termos desta sequência.

Classes

- **Classe**

As classes fornecem um meio de agrupar dados e funcionalidade.

Uma nova classe cria um novo tipo de objeto, permitindo que novas instâncias desse tipo sejam criadas. Cada instância de classe pode ter atributos anexados a ela para manter seu estado. As instâncias de classe também podem ter métodos (definidos por sua classe) para modificar seu estado.

Classes

- **Classe**

As classes fornecem um meio de agrupar dados e funcionalidade.

Uma nova classe cria um novo tipo de objeto, permitindo que novas instâncias desse tipo sejam criadas. Cada instância de classe pode ter atributos anexados a ela para manter seu estado. As instâncias de classe também podem ter métodos (definidos por sua classe) para modificar seu estado.

Introdução ao Python e a Redes Neurais Artificiais

```
exercicio1.py > ...
1 class Pessoa():
2     """docstring for ClassName"""
3     def __init__(self, nome, cpf):
4         super(Pessoa, nome: str).__init__()
5         self.nome = nome
6         self.cpf = cpf;
7
8     def imprimeNome(self):
9         print(self.nome);
10
11    def imprimeCPF(self):
12        print(self.cpf)
13
14    def nomeCPF(self):
15        print(self.nome+ self.cpf)
16
17    def setCpf(self, cpf):
18        self.cpf = cpf;
19
20
21
22 if __name__ == "__main__":
23     x = Pessoa("Tere", "10.10.10.100");
24     x.imprimeNome();
25     x.nomeCPF();
26     x.setCpf("1")
27     x.imprimeCPF();
```

Classes

```
terenciani@terenciani-pc: ~/cursopython
Arquivo Editar Ver Pesquisar Terminal Ajuda
terenciani@terenciani-pc:~/cursopython$ python3 exercicio1.py
Tere
Tere10.10.10.100
1
terenciani@terenciani-pc:~/cursopython$
```

Códigos feitos em Python.

Um Pouco sobre Inteligência Artificial

Motivação

- Visão Computacional.
 - Diversas ferramentas robustas para essa finalidade.
 - Nicho expressivo de desenvolvedores dessa área.

Motivação



Python



Motivação



$$y_{it} = \beta' x_{it} + \mu_i + \epsilon_{it}$$



Problemas

- Detecção.
- Detecção e localização
- Segmentação

Problemas

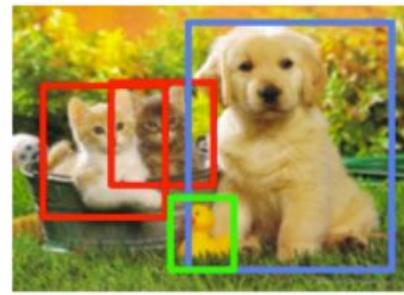
Classificação
de Imagem



Classificação +
Localização



Detecção de
Objetos



Segmentação
de Instâncias



Fonte: [Data Science Academy](#)

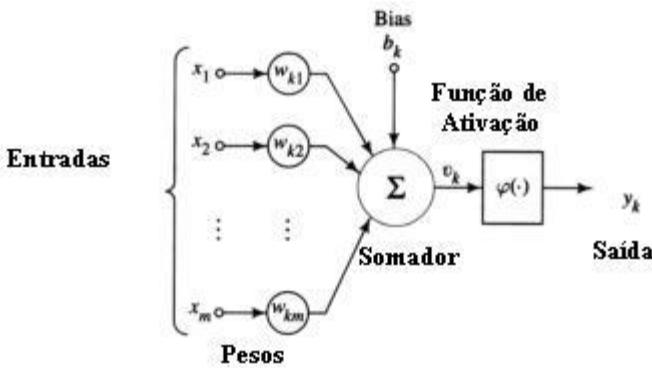
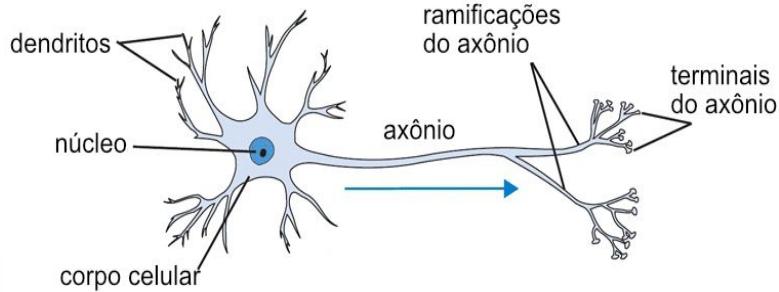
Exemplos

- Detecção/Reconhecimento facial.
- Detecção/Reconhecimento/Segmentação de imagens e objetos.

Redes Neurais Artificiais

- Criadas em 1940.
- Forte inspiração biológica.
- Adormecidas de 1969 a 1980.
- Ideia de interconectar uma malha de neurônios artificiais.
- Aprendizagem supervisionada

Neurônios biológicos vs artificiais



Fonte: [ResearchGate](#)

Fonte: [GSigma UFSC](#)

Redes Neurais Artificiais

- Os sinais são propagados entre os neurônios pelas conexões.
- O objetivo é realizar um processo de aprendizagem modificando os pesos dos neurônios.

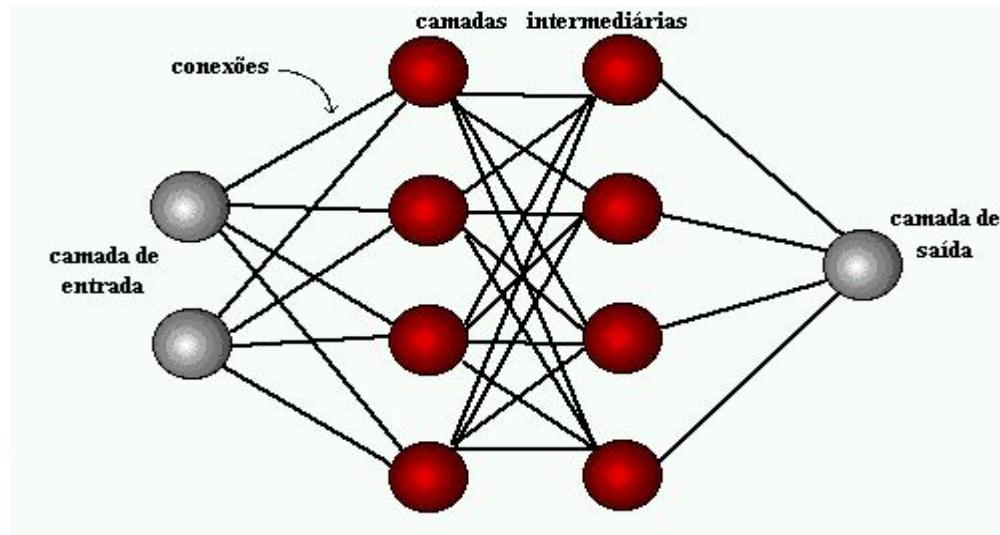
Redes Neurais Artificiais

- Já que o conhecimento da rede está armazenado nos pesos, o objetivo da RNA é encontrar um conjunto de pesos que faça a rede realizar certo comportamento.

Redes Neurais Artificiais

- Os tipos de redes neurais são:
 - *Feed forward* de uma camada.
 - *Feed forward* de Múltiplas camadas.
 - Recorrentes.

Redes Neurais Artificiais



Fonte: [ICMC USP](#)

RNA Perceptron

- Nos algoritmos de reforço há sempre um objetivo a ser alcançado.
- A cada tempo t , uma entrada $x(t)$ (estado do ambiente) leva a rede a uma saída $y(t)$.
- Tentativa e erro e reforço retardado.

RNA Perceptron

- Consiste em uma única camada neurônios e pesos sinápticos.
- Se os dados de entrada forem linearmente separáveis é possível garantir que o algoritmo funcionará.

Algoritmo Perceptron

- Calcular a saída y_i .
- Calcular o erro (valor desejado - obtido).
 - $e_i = d_i - y_i$.
- Atualizar os pesos dos neurônios.
 - $w_i(t+1) = w_i(t) + \alpha * e_i * x_i$
- No qual α representa a taxa de aprendizagem.

Algoritmo Perceptron

- Repetir o processo do slide anterior enquanto $E > 0$ e não atingir um máximo de iterações.

Prática

- Crie uma RNA *Perceptron* para reconhecer a função lógica de conjunção (AND).
- Como modelar o problema?

Prática (Desafio)

- Crie uma RNA *Perceptron* para reconhecer se algumas bolinhas estão acima ou abaixo da diagonal principal da tela [The coding Train](#).

FIM

- THAT'S ALL FOLKS!

Níveis de requisitos

FIM

- Dúvidas?