

# Introducción a la programación

## Usando Python

G. Sebastián Pedersen

Instituto de Industria  
Universidad Nacional de General Sarmiento

Matemática para Economistas III, 1er. cuat. 2019

# Introducción

- ▶ Objetivo: programar un sistema de ecs. difs. y graficarlo.

# Introducción

- ▶ Objetivo: programar un sistema de ecs. difs. y graficarlo.
- ▶ ¿Qué es programar?

# Introducción

- ▶ Objetivo: programar un sistema de ecs. difs. y graficarlo.
- ▶ ¿Qué es programar?
  - ▶ Programar  $\neq$  saber un lenguaje de programación.

# Introducción

- ▶ Objetivo: programar un sistema de ecs. difs. y graficarlo.
- ▶ ¿Qué es programar?
  - ▶ Programar  $\neq$  saber un lenguaje de programación.
  - ▶ Programar  $\neq$  saber usar una computadora.

# Introducción

- ▶ Objetivo: programar un sistema de ecs. difs. y graficarlo.
- ▶ ¿Qué es programar?
  - ▶ Programar  $\neq$  saber un lenguaje de programación.
  - ▶ Programar  $\neq$  saber usar una computadora.
  - ▶ Frase de Edgar Dijkstra: “La Ciencia de la Computación no tiene que ver con las computadoras más que la Astronomía con los telescopios”.

# Introducción

- ▶ Objetivo: programar un sistema de ecs. difs. y graficarlo.
- ▶ ¿Qué es programar?
  - ▶ Programar  $\neq$  saber un lenguaje de programación.
  - ▶ Programar  $\neq$  saber usar una computadora.
  - ▶ Frase de Edgar Dijkstra: “La Ciencia de la Computación no tiene que ver con las computadoras más que la Astronomía con los telescopios”.
- ▶ ¿Qué lenguajes de programación conocen?

# Introducción

- ▶ Objetivo: programar un sistema de ecs. difs. y graficarlo.
- ▶ ¿Qué es programar?
  - ▶ Programar  $\neq$  saber un lenguaje de programación.
  - ▶ Programar  $\neq$  saber usar una computadora.
  - ▶ Frase de Edgar Dijkstra: “La Ciencia de la Computación no tiene que ver con las computadoras más que la Astronomía con los telescopios”.
- ▶ ¿Qué lenguajes de programación conocen?
- ▶ ¿Cuál lenguaje vamos a usar nosotros?



# Introducción

- ▶ Objetivo: programar un sistema de ecs. difs. y graficarlo.
- ▶ ¿Qué es programar?
  - ▶ Programar  $\neq$  saber un lenguaje de programación.
  - ▶ Programar  $\neq$  saber usar una computadora.
  - ▶ Frase de Edgar Dijkstra: “La Ciencia de la Computación no tiene que ver con las computadoras más que la Astronomía con los telescopios”.
- ▶ ¿Qué lenguajes de programación conocen?
- ▶ ¿Cuál lenguaje vamos a usar nosotros?
  - ▶ Respuesta corta: Python.

# Introducción

- ▶ Objetivo: programar un sistema de ecs. difs. y graficarlo.
- ▶ ¿Qué es programar?
  - ▶ Programar  $\neq$  saber un lenguaje de programación.
  - ▶ Programar  $\neq$  saber usar una computadora.
  - ▶ Frase de Edgar Dijkstra: “La Ciencia de la Computación no tiene que ver con las computadoras más que la Astronomía con los telescopios”.
- ▶ ¿Qué lenguajes de programación conocen?
- ▶ ¿Cuál lenguaje vamos a usar nosotros?
  - ▶ Respuesta corta: Python.
  - ▶ Respuesta larga: no importa demasiado. Lo importante son los conocimientos básicos de programación, que son comunes a la mayoría de los lenguajes.

# Recursos Python

## ► Comunidades:

- <http://www.python.org.ar/>
- <https://argentinaenpython.com/>
- <https://twitter.com/pibesdesistemas>
- <https://www.chicasentecnologia.org/>
- <https://twitter.com/lasdesistemas>
- <https://www.meetup.com/Buenos-Aires-Python-Meetup/>
- <https://twitter.com/linuxchixar>

# Recursos Python

## ► Comunidades:

- <http://www.python.org.ar/>
- <https://argentinaenpython.com/>
- <https://twitter.com/pibesdesistemas>
- <https://www.chicasentecnologia.org/>
- <https://twitter.com/lasdesistemas>
- <https://www.meetup.com/Buenos-Aires-Python-Meetup/>
- <https://twitter.com/linuxchixar>

## ► Material, cursos, tutoriales, bibliografía:

- <http://www.python.org.ar/wiki/AprendiendoPython>
- <https://argentinaenpython.com/quiero-aprender-python/aprenda-a-pensar-como-un-programador-con-python.pdf>
- <https://launchpadlibrarian.net/18980633/Python%20para%20todos.pdf>
- Cursos online (en inglés): coursera, datacamp, udemy, Stanford online, edx, codeacademy, Harvard online, etc.

# Recursos Python

## ► Comunidades:

- <http://www.python.org.ar/>
- <https://argentinaenpython.com/>
- <https://twitter.com/pibesdesistemas>
- <https://www.chicasentecnologia.org/>
- <https://twitter.com/lasdesistemas>
- <https://www.meetup.com/Buenos-Aires-Python-Meetup/>
- <https://twitter.com/linuxchixar>

## ► Material, cursos, tutoriales, bibliografía:

- <http://www.python.org.ar/wiki/AprendiendoPython>
- <https://argentinaenpython.com/quiero-aprender-python/aprenda-a-pensar-como-un-programador-con-python.pdf>
- <https://launchpadlibrarian.net/18980633/Python%20para%20todos.pdf>
- Cursos online (en inglés): coursera, datacamp, udemy, Stanford online, edx, codeacademy, Harvard online, etc.

## ► Buscar en internet: hay mucho mucho hecho ya.

# Elementos básicos programación

- ▶ Diferencia entre algoritmo y programa.

# Elementos básicos programación

- ▶ Diferencia entre algoritmo y programa.
- ▶ Herramientas esenciales:
  - ▶ Tipos de datos: enteros, reales, strings, etc.
  - ▶ Variables y expresiones.
  - ▶ Instrucciones: asignación, condicional, ciclo.
  - ▶ Funciones, pasajes de parámetros.

# Elementos básicos programación

- ▶ Diferencia entre algoritmo y programa.
- ▶ Herramientas esenciales:
  - ▶ Tipos de datos: enteros, reales, strings, etc.
  - ▶ Variables y expresiones.
  - ▶ Instrucciones: asignación, condicional, ciclo.
  - ▶ Funciones, pasajes de parámetros.
- ▶ Estructuras de datos:
  - ▶ Listas, arreglos.
  - ▶ Conjuntos, diccionarios.
  - ▶ Pilas, colas.



# Elementos básicos programación

- ▶ Diferencia entre algoritmo y programa.
- ▶ Herramientas esenciales:
  - ▶ Tipos de datos: enteros, reales, strings, etc.
  - ▶ Variables y expresiones.
  - ▶ Instrucciones: asignación, condicional, ciclo.
  - ▶ Funciones, pasajes de parámetros.
- ▶ Estructuras de datos:
  - ▶ Listas, arreglos.
  - ▶ Conjuntos, diccionarios.
  - ▶ Pilas, colas.
- ▶ ¿Cómo se aprende a programar?

# Elementos básicos programación

- ▶ Diferencia entre algoritmo y programa.
- ▶ Herramientas esenciales:
  - ▶ Tipos de datos: enteros, reales, strings, etc.
  - ▶ Variables y expresiones.
  - ▶ Instrucciones: asignación, condicional, ciclo.
  - ▶ Funciones, pasajes de parámetros.
- ▶ Estructuras de datos:
  - ▶ Listas, arreglos.
  - ▶ Conjuntos, diccionarios.
  - ▶ Pilas, colas.
- ▶ ¿Cómo se aprende a programar? Programando... no hay manera de aprender algo sin hacerlo.

# Algoritmo y programa

- ▶ Un *algoritmo* es una secuencia de instrucciones.

# Algoritmo y programa

- ▶ Un *algoritmo* es una secuencia de instrucciones. Por ejemplo:
  1. Moje el cabello.
  2. Coloque champú.
  3. Masajee suavemente y deje actuar por 2 min.
  4. Enjuague.
  5. Repita el procedimiento desde 1.

# Algoritmo y programa

- ▶ Un *algoritmo* es una secuencia de instrucciones. Por ejemplo:
  1. Moje el cabello.
  2. Coloque champú.
  3. Masajee suavemente y deje actuar por 2 min.
  4. Enjuague.
  5. Repita el procedimiento desde 1.
- ▶ Un programa es una implementación de un algoritmo en un lenguaje de programación.

# Algoritmo y programa

- ▶ Un *algoritmo* es una secuencia de instrucciones. Por ejemplo:
  1. Moje el cabello.
  2. Coloque champú.
  3. Masajee suavemente y deje actuar por 2 min.
  4. Enjuague.
  5. Repita el procedimiento desde 1.
- ▶ Un programa es una implementación de un algoritmo en un lenguaje de programación.
  - ▶ El programa representa al algoritmo en el lenguaje.

# Algoritmo y programa

- ▶ Un *algoritmo* es una secuencia de instrucciones. Por ejemplo:
  1. Moje el cabello.
  2. Coloque champú.
  3. Masajee suavemente y deje actuar por 2 min.
  4. Enjuague.
  5. Repita el procedimiento desde 1.
- ▶ Un programa es una implementación de un algoritmo en un lenguaje de programación.
  - ▶ El programa representa al algoritmo en el lenguaje.
  - ▶ Las instrucciones son propias del lenguaje.

```
for i in people.data.users:
    response = client.api.statuses.user_timeline.get(screen_name=i.scre
    print 'Got', len(response.data), 'tweets from', i.screen_name
    if len(response.data) != 0:
        ltdate = response.data[0]['created_at']
        ltdate2 = datetime.strptime(ltdate, '%a %b %d %H:%M:%S +0000 %Y')
        today = datetime.now()
        howlong = (today-ltdate2).days
        if howlong < daywindow:
            print i.screen_name, 'has tweeted in the past', daywindow,
            totaltweets += len(response.data)
            for j in response.data:
                if j.entities.urls:
                    for k in j.entities.urls:
                        newurl = k['expanded_url']
                        urlset.add((newurl, j.user.screen_name))
        else:
            print i.screen_name, 'has not tweeted in the past', daywindi
```

# Make Titles Informative. Use Uppercase Letters.

Subtitles are optional.

- ▶ Use itemize a lot.
- ▶ Use very short sentences or short phrases.



# Make Titles Informative.

You can create overlays. . .

- ▶ using the pause command:
  - ▶ First item.

# Make Titles Informative.

You can create overlays. . .

- ▶ using the `pause` command:
  - ▶ First item.
  - ▶ Second item.
- ▶ using overlay specifications:
- ▶ using the general `uncover` command:

# Make Titles Informative.

You can create overlays. . .

- ▶ using the `pause` command:
  - ▶ First item.
  - ▶ Second item.
- ▶ using overlay specifications:
  - ▶ First item.
- ▶ using the general `uncover` command:

# Make Titles Informative.

You can create overlays. . .

- ▶ using the `pause` command:
  - ▶ First item.
  - ▶ Second item.
- ▶ using overlay specifications:
  - ▶ First item.
  - ▶ Second item.
- ▶ using the general `uncover` command:

# Make Titles Informative.

You can create overlays. . .

- ▶ using the `pause` command:
  - ▶ First item.
  - ▶ Second item.
- ▶ using overlay specifications:
  - ▶ First item.
  - ▶ Second item.
- ▶ using the general `uncover` command:
  - ▶ First item.

# Make Titles Informative.

You can create overlays. . .

- ▶ using the `pause` command:
  - ▶ First item.
  - ▶ Second item.
- ▶ using overlay specifications:
  - ▶ First item.
  - ▶ Second item.
- ▶ using the general `uncover` command:
  - ▶ First item.
  - ▶ Second item.

Make Titles Informative.

Make Titles Informative.



Make Titles Informative.

Make Titles Informative.

Make Titles Informative.

Make Titles Informative.

Make Titles Informative.

Make Titles Informative.

# Summary

- ▶ The **first main message** of your talk in one or two lines.
  - ▶ The **second main message** of your talk in one or two lines.
  - ▶ Perhaps a **third message**, but not more than that.
- 
- ▶ Outlook
    - ▶ Something you haven't solved.
    - ▶ Something else you haven't solved.

# For Further Reading I



A. Author.

*Handbook of Everything.*

Some Press, 1990.



S. Someone.

On this and that.

*Journal of This and That*, 2(1):50–100, 2000.