



Funciones

Clase #10

IIC1103 – Introducción a la Programación

El plan de hoy es...

Clase del lunes está en canvas; códigos en carpeta correspondiente

Primero: menti de repaso

Luego:

- funciones 😊
 - Propias!!

Conozcamos repl.it 😊

f-strings: `print(f"...)`

- Desde Python 3.6
- Se le agrega como prefijo **f**
- Permite combinar variables con strings de forma fácil
- `>>> e1 = 234`
- `>>> v1 = 'Pudahuel'`
- `>>> print(f"En la estacion {v1} el valor es {e1}")`
- **En la estacion Pudahuel el valor es 234**

Problema #0



- Hora de llegada #1? 140301
- Hora de llegada #2? 135949
- Hora de llegada #3? 140159
- ...
- Hora de llegada #n? -1
- El ganador es: #2

Solución

- ```
min_hora = 235959
ganador = 0
corredor = 1
seguir = True
while seguir:
 hora = int(input("Hora de llegada #" + str(corredor) + "? "))
 if hora == -1:
 seguir=False
 else:
 if hora<min_hora:
 ganador=corredor
 min_hora=hora
 corredor+=1
print("El ganador es",ganador)
```

# Problema 1



- Hora de llegada #1? 140301
  - Hora de llegada #2? 135949
  - Hora de llegada #3? 140159
  - ...
  - Hora de llegada #n? -1
- 
- El ganador es: #2
  - Diferencia de tiempo entre primero y último:
  - 192 segundos

# Solución

- ```
min_hora = 235959
max_hora = 0
ganador = 0
corredor = 1
terminar=False
while not terminar:
    hora = int(input("Hora de llegada #" + str(corredor) + "? "))
    if hora == -1:
        terminar=True
    else:
        if hora < min_hora:
            ganador=corredor
            min_hora=hora
        if hora > max_hora:
            max_hora=hora
    corredor+=1
print("El ganador es",ganador)
print("Diferencia de tiempo entre primero y último:")
...

```

Solución

```
min_hora = 235959
max_hora = 0
ganador = 0
corredor = 1
terminar=False
while not terminar:
    hora = int(input("Hora de llegada #" + str(corredor) + "? "))
    if hora == -1:
        terminar=True
    else:
        if hora < min_hora:
            ganador=corredor
            min_hora=hora
        if hora > max_hora:
            max_hora=hora
print("El ganador es",ganador)
print("Diferencia de tiempo entre primero y último:")
s1 = max_hora%100
m1 = (max_hora//100)%100
h1 = (max_hora//10000)%100
segundos_max = s1+m1*60+h1*60*60
s2 = min_hora%100
m2 = (min_hora//100)%100
h2 = (min_hora//10000)%100
segundos_min = s2+m2*60+h2*60*60
print(segundos_max-segundos_min)
```


Solución #2

```
min_hora = 235959
max_hora = 0
ganador = 0
corredor = 1
terminar=False
while not terminar:
    hora = int(input("Hora de llegada #" + str(corredor) + "? ")
    if hora == -1:
        terminar=True
    else:
        if hora < min_hora:
            ganador=corredor
            min_hora=hora
        if hora > max_hora:
            max_hora=hora
print("El ganador es",ganador)
print("Diferencia de tiempo entre primero y último:")
segundos_max = segundos(max_hora)
segundos_min = segundos(min_hora)
print(segundos_max-segundos_min)
```



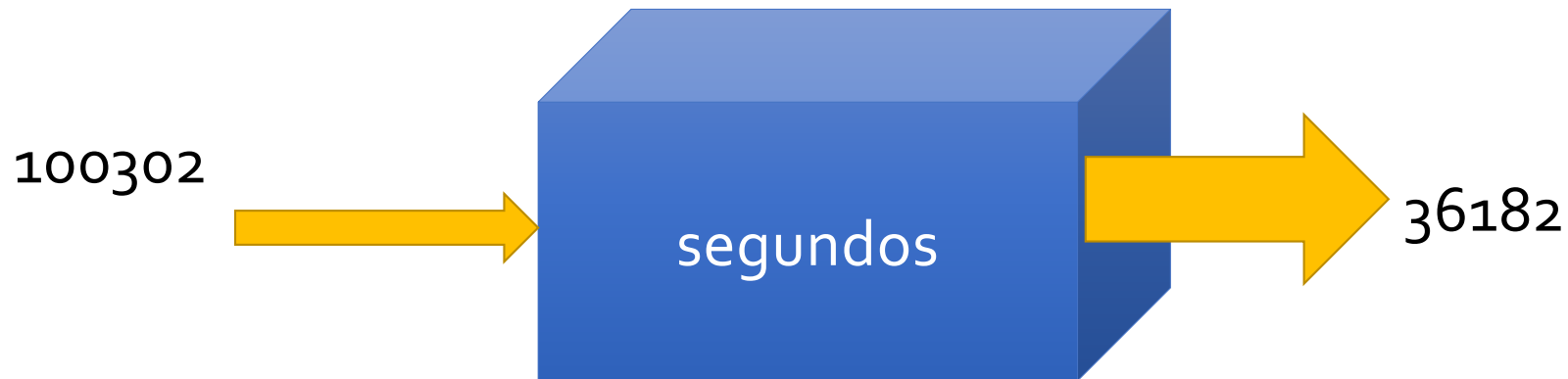
Funciones

- Reciben argumentos/parámetros
- Hacen cálculos
- Retornan/devuelven algún valor



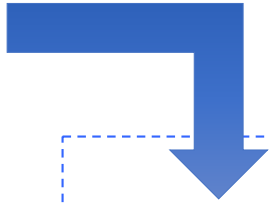
Funciones

- Reciben argumentos/parámetros
- Hacen cálculos
- Retornan/devuelven algún valor



Dentro de la función...

hora



segundos

Instrucciones usando hora
+ cualquier otra
instrucción/cálculo
necesario

¿Cómo sabremos cuál es el resultado?

Dentro de la función...

hora

segundos

Instrucciones usando hora
+ cualquier otra
instrucción necesaria

return resultado

resultado



```
graph TD; hora --> Box; segundos --> Box; subgraph Box [ ]; I[Instrucciones usando hora + cualquier otra instrucción necesaria]; end; Box --> resultado; return[return resultado];
```

The diagram illustrates the flow of data into and out of a function. Two inputs, 'hora' and 'segundos', are shown as text labels. A large blue arrow originates from the 'hora' label, turns right, and then points down into a dashed blue rectangular box. Another blue arrow points from the 'segundos' label into the same box. Inside the box, the text 'Instrucciones usando hora + cualquier otra instrucción necesaria' is written in blue. Below this text, the text 'return resultado' is written in blue. A large blue arrow points from the bottom of the dashed box to the word 'resultado' at the bottom of the diagram.

Definición de función segundos

- **def** segundos(hora):

```
    s = hora%100
```

```
    m = (hora//100)%100
```

```
    h = (hora//10000)%100
```

```
    return s+m*60+h*60*60
```


Dentro de la función...

definir
función

nombre de la función

parámetros/argumentos

• `def segundos(hora):`

• 

```
s = hora%100  
m = (hora//100)%100  
h = (hora//10000)%100  
return s+m*60+h*60*60
```

• variable **LOCAL!**

- instrucciones internas a la función (ojo: deben estar indentadas)
- retornar resultado (con instrucción return)

Estructura de un programa completo: ojo con la indentación

```
import math
def area_circulo(r):
    return math.pi*r*r
def area_triangulo(a,b,c):
    s=(a+b+c)/2
    return math.sqrt(s*(s-a)*(s-b)*(s-c))
a=float(input("a? "))
b=float(input("b? "))
c=float(input("c? "))
radio = max(a,b,c)/2
print("area = "+str(area_circulo(radio)-area_triangulo(a,b,c)))
```

instrucciones import

definiciones de funciones
que usaremos más abajo

programa principal:
llamadas a funciones

Nombres de funciones

- Similar a variables: letras, números, algunos símbolos: primer carácter NO puede ser un número
- No usar *keywords* reservados de python
- No usar mismo nombre que variable

Funciones: ¿por qué usarlas?

- Dividir para conquistar
- Separar trabajo entre programadores
- Programas más legibles y comprensibles
- Eliminar código repetitivo. Si quiero hacer un cambio, lo hago en un solo lugar.
- Realizar trabajos más complejos con operaciones compuestas
- Pueden ser útiles para varios programas
 - ... pensar, por ejemplo, en funciones disponibles en math!

Problema #2

- Escribe una función `sumatoria(n)` que entregue la suma de los números desde 1 a `n`. Si `n` es negativo, debe entregar 0.
- Ejemplos:
 - `x = sumatoria(4)` #x vale 10
 - `y = sumatoria(-4)` #y vale 0

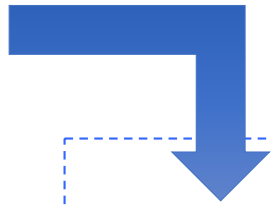
Funciones

- Reciben argumentos/parámetros
- Hacen cálculos
- Retornan/devuelven algún valor



Dentro de la función...

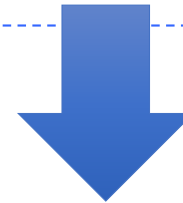
x



sumatoria

Instrucciones usando x
+ cualquier otra
instrucción necesaria

return resultado



resultado

Dentro de la función...

definir

función

nombre de la función

parámetros/argumentos

• `def sumatoria(x):`

- ```
s = 0
for i in range(1,x+1):
 s+=i
return s
```

- variable **LOCAL!**

- instrucciones internas a la función (ojo: deben estar indentadas)
- retornar resultado (con instrucción `return`)

### Problema 3: 🦎



# Resumen de hoy

definir  
función

nombre de la función

parámetros/argumentos

```
• def area_triangulo(a,b,c):
```

```
 s=(a+b+c)/2
```

```
 return math.sqrt(s*(s-a)*(s-b)*(s-c))
```

• variable **LOCAL**!

- instrucciones internas a la función (ojo: deben estar indentadas)
- retornar algo (con instrucción return)

\* Funciones pueden retornar o no

\* Al ejecutar "return", se sale de la función y vuelve al lugar donde fue llamada