Introducción a la programación

Temario

- Motivación
- Conceptos básicos:
  - Lenguaje de programación
  - Programa
  - Algoritmo
  - Variables
  - Tipos de datos
  - Expresiones
- Instrucciones
  - Asignación
  - Entrada / Salida
  - Estructuras de control
    - Selección
    - Iteración
- Subprogramas (funciones)

Motivación

- ¿Por qué robótica?:
  - Motiva y despierta la curiosidad en los niños.
  - Promueve la experimentación.
  - Incentiva la imaginación y la creatividad.
  - Da un marco en el que abordar temas de física, álgebra, geometría, electrónica y programación, entre otros.
  - Brinda seguridad y confianza en aspectos tecnológicos
  - Promueve el paradigma “Edutainment”: hacer entretenido el aprendizaje.

Lenguaje de programación

Pascal, C, C++, Java, Python, Prolog, Haskell, ...

Permiten dar instrucciones a la computadora con un cierto nivel de abstracción, es decir, no operan directamente sobre los bits.

- lenguajes compilados: el código escrito por el programador se compila (traduce) para generar un código ejecutable por la computadora.
- lenguajes interpretados: el código escrito por el programador va siendo ejecutado directamente por un intérprete. (ej.: Python)

Lenguaje de programación

- Para todos los ejemplos y ejercicios de este curso se utiliza el lenguaje Python (http://python.org/).
- IDE Web: http://ideone.com/
- Terminal [+ Editor]
  - GEdit, python
- Pippy
  - Manual de instalación en el entorno virtual

Programa

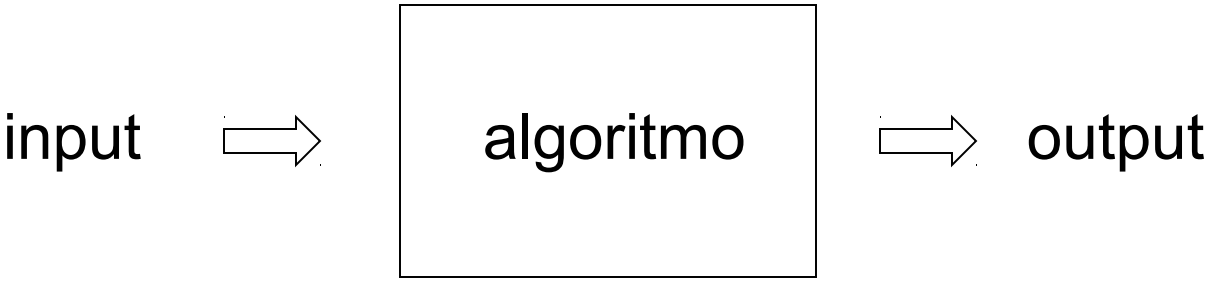
Secuencia de instrucciones que una computadora puede interpretar y ejecutar.

Escrito en un lenguaje de programación.

Se manipulan datos y se ejecuta un algoritmo.

Programa

Para ciertos datos de entrada (input) el programa aplica un algoritmo y genera una salida (output).



Los algoritmos son el objeto de estudio de la programación.

Algoritmo

(Real Academia Española. www.rae.es)

algoritmo.

1. m. Conjunto ordenado y finito de operaciones que permite hallar la solución de un problema.

- Características
  - Preciso: orden en que se realizan los pasos.
  - Definido: siempre se obtiene el mismo resultado sin importar el nro. de veces que se aplique.
  - Finito: tiene fin.

Algoritmo

(Real Academia Española. www.rae.es)

algoritmo.

1. m. Conjunto ordenado y finito de operaciones que permite hallar la solución de un problema.

¿Cuantas líneas se necesitan para escribir el algoritmo?

- Características
  - Preciso: orden en que se realizan los pasos.
  - Definido: siempre se obtiene el mismo resultado sin importar el nro. de veces que se aplique.
  - Finito: tiene fin.

¿Cuanto demora en terminar?

Algoritmo

Ejemplo: Receta para hacer una torta de chocolate

Entrada: 100 g manteca, 150 g azúcar, 2 huevos, 1 cta vainilla, 3 cdas cacao, 2.5 tazas harina, 4 ctas polvo de hornear, ½ cta sal, ¾ tazas leche

Algoritmo: Batir la manteca y el azúcar juntos.

Agregar los huevos de a uno, sin dejar de batir.

Agregar el cacao y mezclar de a poquito la harina, la sal, el polvo de hornear y la leche.

Verter en una tortera previamente enmantecada y enharinada.

Hornear durante 35 minutos a 180 grados.

Desmoldar en un plato de torta.

Salida: Torta de chocolate

Algoritmo

Ejemplo:

Calcular la suma de los salarios de una lista de empleados.

Entrada:	lista de empleados con diferentes datos incluyendo salario
Algoritmo:	anotar el valor 0 recorrer la lista sumando cada salario al valor anotado
Salida:	valor final anotado

Algoritmo

- El algoritmo para la suma de salarios resuelve muchas instancias diferentes, es decir, funciona correctamente para diferentes datos de entrada.
  - ¿Cualquier dato de entrada es válido?
    - lista vacía
    - lista con salarios negativos
    - lista con todos 0
- El algoritmo para preparar la torta siempre tiene la misma entrada y ¿la misma salida?
  - ¿Se puede generalizar?

Algoritmo

Ejercicios

Escribir algoritmos que resuelvan los siguientes problemas:

1)Escribir un algoritmo que indique el procedimiento a seguir por un lavarropas.

2)Modificar el algoritmo para la suma de salarios de modo que, además de dar como salida la suma total, dé la cantidad de salarios procesados.

3)Indicar los pasos a seguir para que un cajero automático determine la cantidad de billetes que debe entregar a un usuario, dado un monto que este solicita. El cajero tiene billetes de $1000, $500, $200 y $100.

Programa

Ejemplo de programa en Python

Calcular el perímetro de un triángulo equilátero

```
# cálculo del perímetro de un triángulo equilátero
lado = input('Lado del triángulo: ')
perim = lado * 3
print 'Perímetro: ', perim
```

Variables

Una variable es un espacio de memoria de la máquina donde se pueden almacenar y consultar valores.

Atributos

- Nombre
- Tipo
- Valor
- Alcance

- ¿Qué variables se necesitan para los algoritmos vistos anteriormente?

Variables

Los nombres de variables se llaman identificadores.

Pueden contener letras, números y guión bajo; no pueden empezar por un número.

Mayúsculas y minúsculas son letras diferentes.

Es importante usar nombres mnemotécnicos.

Ejemplos válidos: num1, num_entrada, potencia, resultado
Ejemplos no válidos: 1num, número
num1, Num1, NUM1 son identificadores ***diferentes.***

¿Los valores de una variable Python siempre deben variar?

Tipos de datos básicos

- Datos numéricos
  - enteros
  - reales
- Cadenas de caracteres
- Lógicos / “Booleanos”
  - dos valores: True y False
  - muy importantes para controlar el flujo del programa

Tipos de datos básicos (en python)

- Datos numéricos

  Enteros $\Rightarrow$ int 0, 5, 908, -90

  Reales $\Rightarrow$ float 0.0, 5.0, 5.9876, 6.02e23

  - Ejemplos:
    - » Carga del electrón en coulomb: -1.602e-19
    - » 1 mol: 6.02e23
    - » Personas en el mundo: 7e9

Tipos de datos básicos (en python)

- Cadenas de caracteres

  Cadenas $\Rightarrow$ str 'a', “mensaje”, 'resultado final: '

  - Hay cadenas más largas:

    '''Esta es una
    cadena que tiene
     3 líneas'''

  - Pueden delimitarse por comillas o apóstrofes

Tipos de datos básicos (en python)

- Lógicos / “Booleanos”

  Lógicos $\Rightarrow$ bool True, False

  - Una condición verdadera es True
  - Una condición falsa es False

Tipos de datos básicos

¿Qué tipos le corresponden a los siguientes valores?

12980

12980.0

12980.567

True

'True'

'?'

TRUE

Tipos de datos básicos

¿Qué tipos le corresponden a los siguientes valores?

12980	int
12980.0	float
12980.567	float
True	bool
'True'	str
'?'	str
TRUE	ninguno!!

Expresiones

Se pueden formar expresiones aplicando operadores a diferentes valores constantes o utilizando variables:

```
3 + 4
5.8 * (2 + 6)
valor1 + valor2 / valor3
'a' + 'b' + 'c'
True and False
(num1 < num2 ) and (num3 = = 0)
```

A las expresiones les corresponde un tipo que depende de los valores intervinientes y de los operadores aplicados

Expresiones

Operadores

operador	descripción
or	el “o” de dos valores bool, retorna un bool
and	el “y” de dos valores bool, retorna un bool
not	negación de un bool, retorna un bool
< <= > >= != ==	comparación de expresiones, retorna un bool
+ -	suma y resta de enteros o reales
* / %	producto, cociente y resto de enteros o reales
**	potenciación

Algunos operadores se aplican también sobre datos de tipo str:
la expresión 'ho' + 'la' es equivalente al string 'hola'

Expresiones

Operadores

AND		
A	B	Salida
False	False	False
False	True	False
True	False	False
True	True	True

Es verdadero sólo si ambos valores lo son.

OR		
A	B	Salida
False	False	False
False	True	True
True	False	True
True	True	True

Es verdadero si algún valor lo es.

NOT	
A	Salida
False	True
True	False

Es verdadero cuando el valor es falso.

Expresiones

¿Qué tipos les corresponden a las siguientes expresiones?

3 + 4

5.8 * (2 + 6)

valor1 + valor2 / valor3

'a' + 'b' + 'c'

True and False

(num1 < num2) and (num3 = = 0)

Expresiones

¿Qué tipos les corresponden a las siguientes expresiones?

3 + 4 **int**

5.8 * (2 + 6) **float**

val1 + val2 / val3

depende de los valores de las variables:

- si alguna es float, la expresión es **float**
- si todas son int, la expresión es **int**

para que la división dé un resultado real, alguno de los argumentos debe ser real (por ejemplo: 5.0)

'a' + 'b' + 'c' **str**

True and False **bool**

(num1 < num2) and (num3 = = 0) **bool**

Instrucciones

- Asignación
- Entrada / Salida
- Estructuras de control
  - selección
  - iteración

Instrucciones: Asignación

El símbolo **=** permite asignar un valor a una variable.

La instrucción

edad = 8

se lee: “a edad le asigno el valor 8”

Importante: no confundir con la comparación!
La expresión

edad = = 8

chequea si edad es igual a 8

Instrucciones: Asignación

Al hacer

edad = 8

se escribe el valor 8 en el espacio de memoria correspondiente a la variable edad.

Si luego hago

edad = edad + 2

se escribe el valor 10 en ese mismo lugar, por lo que se sobrescribe el 8.

Instrucciones: Asignación

- Notar que del lado izquierdo del = siempre hay una variable.

- Del lado derecho siempre hay un valor. puede ser una expresión simple o compuesta.
  - salir = True
  - valor1 = valor2
  - valor1 = (num+1) / num * 2
  - max = funcion_maximo(valor1, valor2)

Instrucciones: Entrada / Salida

print permite desplegar mensajes o datos en la pantalla:

La instrucción: `print 'Comienza el programa'`
despliega el texto que está entre comillas en la pantalla.

La instrucción: `print resultado`
despliega el valor de la variable resultado en la pantalla

La instrucción: `print 'resultado=', result`
despliega el texto “resultado=“ seguido del valor de result

Instrucciones: Entrada / Salida

¿Qué despliega la instrucción?

```
print 'resultado 1=', res1, ', resultado 2=', res2
```

Instrucciones: Entrada / Salida

input( ) permite que el usuario ingrese datos desde el teclado

La instrucción

valor_entrada = input( )

espera que el usuario ingrese algún valor y lo asigna a valor_entrada.

El valor ingresado puede ser de cualquier tipo:

189 (int)

45.98 (float)

'hola esta es una cadena de caracteres' (str)

True (bool)

Instrucciones: Entrada / Salida

Además de leer un dato de la entrada, input( ) permite desplegar un mensaje o dato, que se le pasa como parámetro:

La instrucción

```
valor1 = input('Ingrese un valor entero: ')
```

despliega el texto entre comillas simples, espera que el usuario ingrese algún valor y lo asigna a valor1.

Es lo mismo que poner:

```
print 'Ingrese un valor entero: ',
valor1 = input()
```

Instrucciones: Entrada / Salida

raw_input( ) permite que el usuario ingrese cadenas desde el teclado, sin necesidad de escribir comillas

La instrucción

nombre = raw_input(“¿Cuál es su nombre?”)

espera que el usuario ingrese algún valor y lo asigna a nombre.

Siempre devuelve una cadena.

Es más “seguro” que input.

Ejercicios (asignación, entrada/salida)

1) Escribir un programa para calcular el perímetro de un triángulo cualquiera.

2) Escribir un programa que lea dos mensajes ingresados por el usuario y despliegue la concatenación de los dos.

3) Escribir un programa que calcule y despliegue la suma y el promedio de dos enteros ingresados por el usuario.

4) Escribir un programa que lea dos valores enteros y despliegue el cociente y el resto de la división entera entre ellos.

Estructuras de control

- En los ejemplos vistos hasta ahora, las instrucciones se ejecutan en forma secuencial.
- Se empieza por la primera instrucción y se continúa en orden hasta la última.
- Todas las instrucciones son ejecutadas.
- Ninguna instrucción se ejecuta más de una vez.

Estructuras de control

- Las estructuras de selección e iteración permiten modificar la ejecución del programa.

- **Selección**: se chequean condiciones para decidir qué instrucciones ejecutar. Esto implica que algunas instrucciones pueden no ejecutarse.

- **Iteración**: algunas instrucciones se ejecutan varias veces.