Conceptos básicos

Introducción a la Programación

Departamento de Ciencias Básicas, UNLu

Hasta ahora sabemos:

¿Qué es un algoritmo?

Conjunto de instrucciones o reglas definidas y no-ambiguas, ordenadas y finitas que permite, solucionar un problema, y llevar a cabo una tarea.

¿Qué es un programa de computación?

Es una secuencia de instrucciones que le indican a una computadora cómo realizar una tarea dada.

¿Qué es un programador?

Es la persona que escribe la secuencia de instrucciones.

Hasta ahora sabemos:

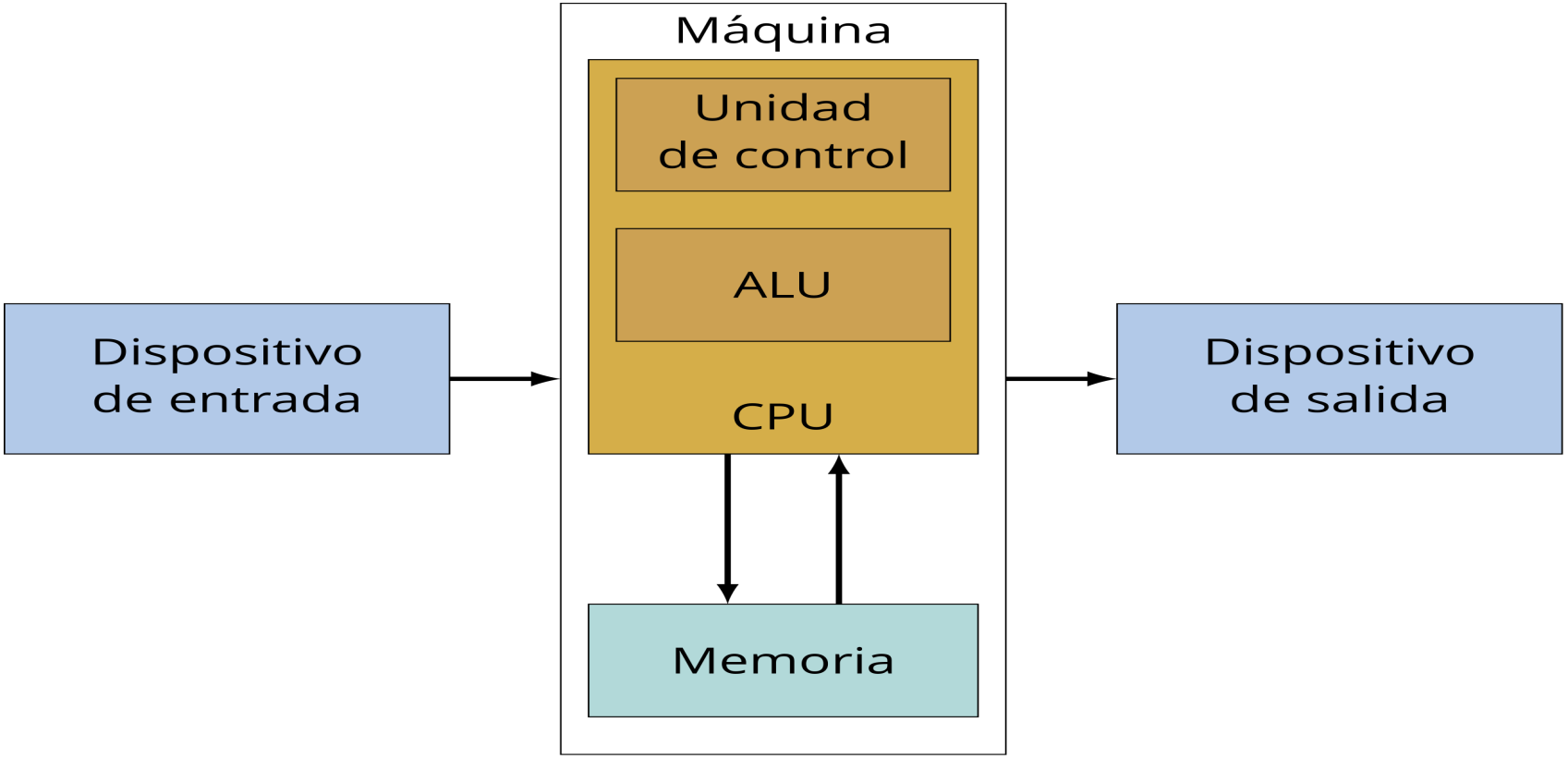
¿Qué es una computadora?:

Una máquina que almacena y procesa información bajo el control de un programa que puede cambiar.

Llamamos a este proceso cómputo.

Computadora: arquitectura de von Neumann

Computadora: máquina que almacena y procesa información bajo el control de un programa que puede cambiar. En la memoria almacena los programas y la información, y en la CPU (Central Processing Unit) hace el procesamiento.



Lenguaje humano

Una característica distintiva de los humanos respecto de los otros seres vivos es que tenemos lenguaje. El lenguaje es la base de nuestro pensamiento.

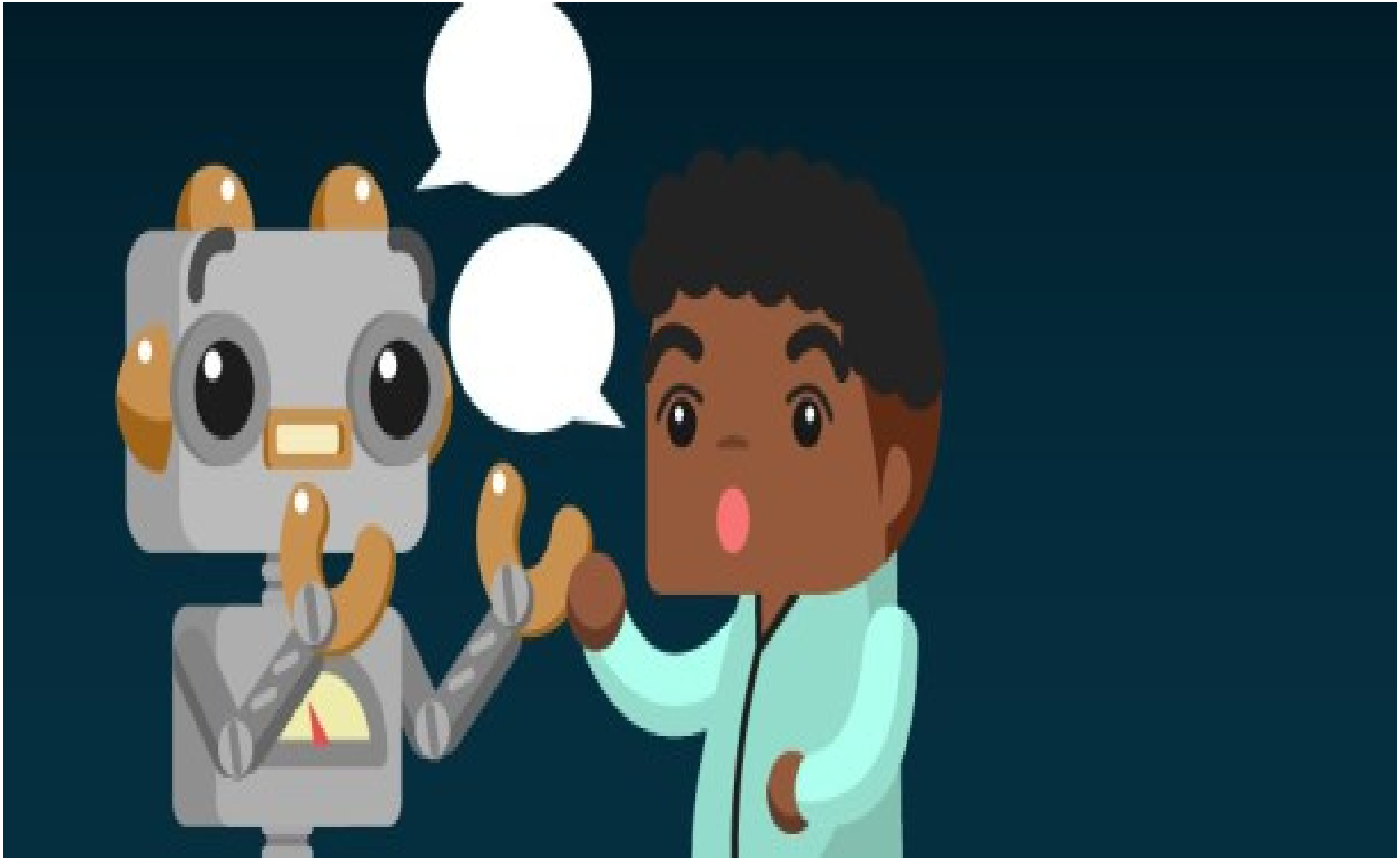
Somos capaces de abstraer la realidad y le ponemos nombres usando nuestro lenguaje.

Nuestro lenguaje nos permite también planificar tareas.

Los lenguajes humanos tienen miles de palabras distintas, y reglas sintácticas para combinarlas.

A una sentencia escrita en un lenguaje humano, le decimos que está escrita en lenguaje natural.

Humanos y Computadoras: similitudes y diferencias



Humanos y Computadoras: similitudes y diferencias

Humanos	***Computadoras***
Plan de tarea: Algoritmos	Plan de tarea: Programas
Lenguaje natural: ambiguo (una sentencia podría tener múltiples interpretaciones)	Lenguaje de programación: no ambiguo (una sentencia tiene una sola interpretación)
Los lenguajes humanos tienen un vocabulario, símbolos (por ej, dos puntos ':') y reglas sintácticas para combinar los vocablos.	Los lenguajes de programación tienen un vocabulario para armar estructuras y hacer operaciones, símbolos, y reglas sintácticas. Pueden tener también librerías (o módulos) que amplían su vocabulario.
Nuestro mundo es complejo. Abstraemos la complejidad del mundo en diferentes clases, lo que nos permite saber cómo interactuar con el mundo.	Su mundo son los datos. Clasifican los datos en *Tipos de Datos*, lo que les permite saber las operaciones válidas para cada tipo de dato, y de esa forma interactuar con los datos.
Tenemos 5 sentidos que le mandan información a nuestro cerebro.	Pueden tener sensores (por ej, cámara) y todo lo transforman en input (datos)
Podemos entender instrucciones ambiguas e inferir cuál es la intención de una tarea que nos piden hacer.	Ejecutan exactamente las instrucciones del programa. **No infieren la intención del programador**.

Ejemplo: vocabulario del lenguaje Python para armar estructuras (keywords o palabras reservadas)

```
rosana@cerebro:~$ python3
Python 3.6.10 |Anaconda, Inc.| (default, Jan  7 2020, 21:14:29)
[GCC 7.3.0] on linux
Type "help", "copyright", "credits" or "license" for more information.
>>> help()

Welcome to Python 3.6's help utility!

If this is your first time using Python, you should definitely check out
the tutorial on the Internet at https://docs.python.org/3.6/tutorial/.

Enter the name of any module, keyword, or topic to get help on writing
Python programs and using Python modules.  To quit this help utility and
return to the interpreter, just type "quit".

To get a list of available modules, keywords, symbols, or topics, type
"modules", "keywords", "symbols", or "topics".  Each module also comes
with a one-line summary of what it does; to list the modules whose name
or summary contain a given string such as "spam", type "modules spam".

help> keywords

Here is a list of the Python keywords.  Enter any keyword to get more help.

False               def                 if                  raise
None                del                 import              return
True                elif                in                  try
and                 else                is                  while
as                  except              lambda              with
assert              finally             nonlocal            yield
break               for                 not
class               from                or
continue            global              pass
```

Ejemplo: símbolos del lenguaje Python

```
help> symbols

Here is a list of the punctuation symbols which Python assigns special meaning
to. Enter any symbol to get more help.

!=                  +                   <=                  __
"                   +=                  <>                  `
"""                 ,                   ==                  b"
%                   -                   >                   b'
%=                  -=                  >=                  f"
&                   .                   >>                  f'
&=                  ...                 >>=                 j
'                   /                   @                   r"
'''                 //                  J                   r'
(                   //=                 [                   u"
)                   /=                  \                   u'
*                   :                   ]                   |
**                  <                   ^                   |=
**=                 <<                  ^=                  ~
*=                  <<=                 _
```

Ejemplo: librerías (o módulos) del lenguaje Python

```
help> modules

Please wait a moment while I gather a list of all available modules...

OpenSSL             audioop             ipaddress           setuptools
PIL                 base64              itertools           shelve
PyQt5               bdb                 json                shlex
__future__          binascii            keyword             shutil
_ast                binhex              kiwisolver          signal
_asyncio            bisect              lib2to3             sip
_bisect             bottleneck          libarchive          sipconfig
_blake2             bs4                 linecache           sipdistutils
_bootlocale         builtins            locale              site
_bz2                bz2                 logging             six
_cffi_backend       cProfile            lzma                skimage
_codecs             calendar            macpath             smtpd
_codecs_cn          certifi             macurl2path         smtplib
_codecs_hk          cffi                mailbox             sndhdr
_codecs_iso2022     cgi                 mailcap             socket
_codecs_jp          cgitb               marshal             socketserver
_codecs_kr          chardet             math                socks
_codecs_tw          chunk               matplotlib          sockshandler
```

Ejemplo: sintaxis y semántica del lenguaje Python

docs.python.org/3/reference/index.html

Python » English 3.8.2 Documentation »

The Python Language Reference

This reference manual describes the syntax and “core semantics” of the language. It is terse, but attempts to be exact and complete. The semantics of non-essential built-in object types and of the built-in functions and modules are described in The Python Standard Library. For an informal introduction to the language, see The Python Tutorial. For C or C++ programmers, two additional manuals exist: Extending and Embedding the Python Interpreter describes the high-level picture of how to write a Python extension module, and the Python/C API Reference Manual describes the interfaces available to C/C++ programmers in detail.