INTRODUCCIÓN A LA PROGRAMACIÓN

1º Bachillerato

¿Qué es un programa?

- Componente informático que se utiliza para realizar una determinada tarea.

- Se compone de algoritmos y estructuras de datos
  - Algoritmos: Secuencia de instrucciones cuyo objetivo es solucionar un problema
  - Estructura de datos: Elementos para almacenar la información que necesita el programa.

- Tipos:
  - Gráficos: El usuario interactúa con el programa mediante ventanas, menús, botones: Word, Paint…
  - Textuales: El usuario interactúa con el programa tecleando ordenes y comandos: Programas que se ejecutan en un terminal

Conceptos importantes

- **Lenguaje de programación: "Idioma" en el que se escriben los programas. Cada lenguaje tiene sus propias instrucciones.**

  Ejemplos: C, C++, Java, Python…

- **Compilador: Programa que se encarga de traducir el programa a código entendibles por el ordenador.**

- **IDE (Entorno Integrado de Desarrollo): Programa que se usa para escribir código en un determinado lenguaje. Contiene:**
  - **Editor de texto.**
  - **Compilador incorporado.**
  - **Opciones de ayuda y asistencia a la hora de escribir el programa.**

Fases en la creación de un programa

Formalización: Especificación del problema que queremos resolver.

Análisis: Plantear la solución al problema

Codificación: Elección de un lenguaje de programación, y escritura de las instrucciones del programa.

Compilación: Traducción a código máquina

Ejecución: Probar el programa

Evaluación: Se valora si el programa hace lo que tiene que hacer.

Python

- Lenguaje de programación creado en los años 90.
- Multiplataforma: Funciona en cualquier sistema operativo.
- Gran comunidad de desarrolladores
- Principales usos:
  - Educación
  - Automatización industrial
  - Gestión de redes
  - Desarrollo web
- IDE utilizado: IDLE, Eclipse, PyDev,…

Python: Tipos de datos

- Numéricos
  - Enteros: 23, 345678, -168
  - Reales: 34.87959, -0.45
- Cadenas de texto: Se representan entre comillas
  - “miguel angel”, “56780034-x”
- Booleanos: True, False

Python: Variables

- Identificador que sirve para almacenar datos de cualquiera de los tipos existentes.

Ejemplos:

```
num = 34.334
edad = 16
calle = “Avenida Maria Luisa”
mayorEdad = True
```

Python: Comunicación con el usuario

- **Instrucción que imprime algo en la pantalla: print**
  - **Para enlazar en la cadena que imprimimos el valor de alguna variable, se usa el operador %d (variable tipo entera, %f (variable tipo real), %s (variable tipo cadena)**

- **Instrucción que lee lo que teclea el usuario:**
  - **Si el dato introducido es numérico: input()**
  - **En otro caso: raw_input()**

Ejemplo 1: Programa que muestre mensaje que conste del nombre del colegio, la calle y su población.

```
print “Salesianos San Luis Rey”
print “Avenida Maria Auxiliadora”
print “Palma del Rio”
```

Python: Comunicación con el usuario

Ejemplo 2: Programa que pida al usuario su nombre, y lo muestre por pantalla.

```
print “Por favor, dime tu nombre”
nombre = raw_input()
print “Gracias, tu nombre es %s” %nombre
```

Ejemplo 3: Programa que pida al usuario un número, y lo muestre por pantalla

```
print “Por favor, dame un numero”
numero = input()
print “El numero que has introducido es %d” %numero
```

Ejercicios

- **Programa que muestre por pantalla lo siguiente:**

```
Juan Garcia Peñalver
Estudiante de Bachillerato
```

- **Programa que pida al usuario su nombre y su dni, y muestre un mensaje en pantalla con la información introducida**

- **Programa que pida al usuario dos números y los muestre por pantalla.**

Python: Operaciones

- **Aritméticas**:
  Suma, Resta, Multiplicación, División: +,-,*,/
  Resto de división: %
  Potencia: **

Si hacemos una operación con dos números enteros, el resultado será ENTERO. Si alguno de los datos es real, el resultado será REAL

Ejemplo

```
num1 = 2
num2 = 3
num3 = 2.5
resultado1 = num1 + num2 # Daría 5
resultado2 = num2 ** num1 # Daría 9
resultado3 = num2 % num1 # Daría 1
resultado4 = num3 * num2 # Daría 7.5
```

Python: Operaciones

- **Comparación**

  > (Mayor), < (Menor), >= (Mayor o igual), <= (Menor o igual), == (iguales), != (distintos)

Ejemplo

```
num1 = 2
num2 = 4
num1 > num2 # FALSO
num1 < num2 # VERDADERO
num1 == num2 # FALSO
num1 != num2 # VERDADERO
```

Ejercicio

- **Programa que pida al usuario dos números enteros, y muestre por pantalla el cociente de la división de los números y el resto.**

Ejemplo ejecución:

Dame el dividendo

14

Dame el divisor

5

El resultado de dividir 14 y 5 es 2, y el resto es 4.

Python: Estructura selectiva

- Hasta ahora solo hemos realizado programas lineales: se ejecuta una línea tras otra secuencialmente.
- Estructura selectiva: Se ejecuta un bloque de código u otro en función de una condición.
  Funcionamiento:
  - Se evalúa una expresión.
  - Según el resultado de la evaluación, se ejecuta un conjunto de instrucciones u otro.

Python: Estructura selectiva

- **Forma simple:**

```
if condición:
        instrucción 1
        …
        instrucción n
else:
        instrucción 1
        …
        instrucción n
```

```
num = 2
if num >= 5:
        print “Mayor o igual que 5”
else:
        print “Menor que 5”
```

Ejercicio

- **Programa que pida un número al usuario y muestre por pantalla si es par o impar.**

Ejemplo ejecución:

Dame un numero

14

El número es par

Python: Estructura selectiva

- **Forma múltiple:**

```
if condición 1:
        instrucción 1
        …
        instrucción n
elif condición2:
        instrucción 1
        ….
        instrucción n
…..
else:
        instrucción 1
        ….
        instrucción n
```

```
num = 2
if num < 0:
        print “Numero positivo”
elif num > 0:
        print “Numero negativo”
else:
        print “Numero igual a 0”
```

Ejercicio

- **Programa que pida un numero al usuario, y muestre un mensaje indicando si el número es positivo, negativo o cero.**

Ejemplo ejecución:

Dame un numero

14

El número es par

Python: Estructura selectiva

- Comprobar varias condiciones
  - Para enlazar varias comprobaciones usamos los operadores lógicos:
    - and: sirve para enlazar dos condiciones. El resultado será verdadero si ambas condiciones son verdaderas.
    - or: sirve para enlazar dos condiciones. El resultado será verdadero si alguna de las condiciones es verdadera.
    - not: sirve para negar una condición. El resultado será verdadero si la condición original era falsa.

Ejercicio

- **Programa que pida las notas del examen de matemáticas y de historia, y nos muestre por pantalla si hemos aprobado los dos exámenes, si hemos suspendido alguno, o si hemos suspendido los dos.**

Ejemplo ejecución:

Dime la nota del examen de matemáticas

3

Dime la nota del examen de historia

6

Has suspendido uno de los dos exámenes.

Python: Estructura repetitiva. WHILE

- **Comprueba una condición, y mientras esta condición se cumpla, ejecutará el bloque de código que tenga asociado.**

```
while condición:
    instrucción 1
    …
    instrucción n
```

```
n = 0
while n < 5:
    print n
    n = n+1
```

Ejercicio

- **Programa que pida al usuario un número entero positivo y realice la suma desde 0 hasta ese número.**

  Ejemplo ejecución:

  Dame un numero entero positivo

  5

  La suma desde 0 hasta 5 es 15