

Tarea 1

Unidad1: Algoritmos de Ordenamiento.

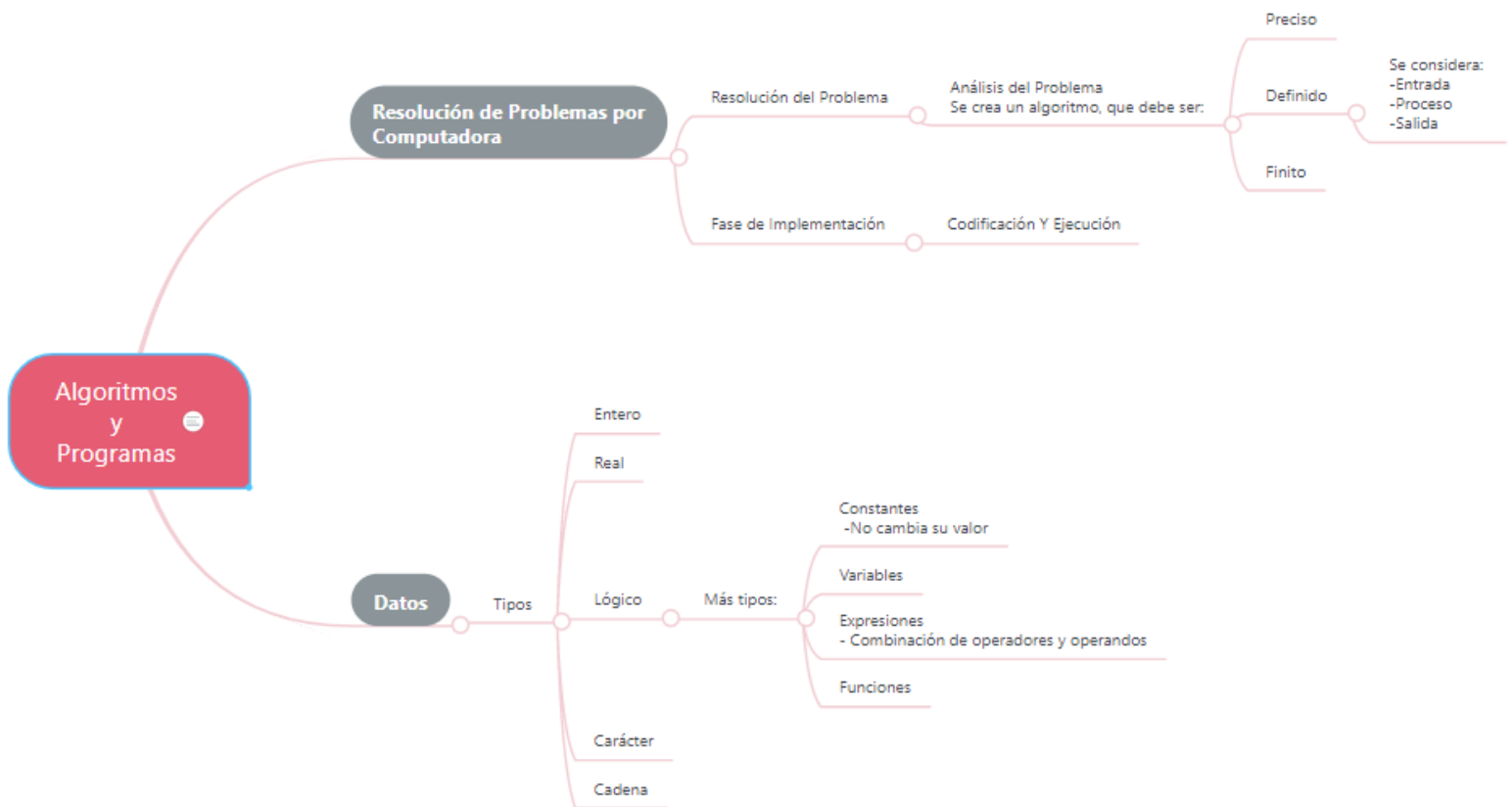
Profesor. Gerardo Tovar Tapia.

Alumna: Monroy Velázquez Alejandra Sarahí

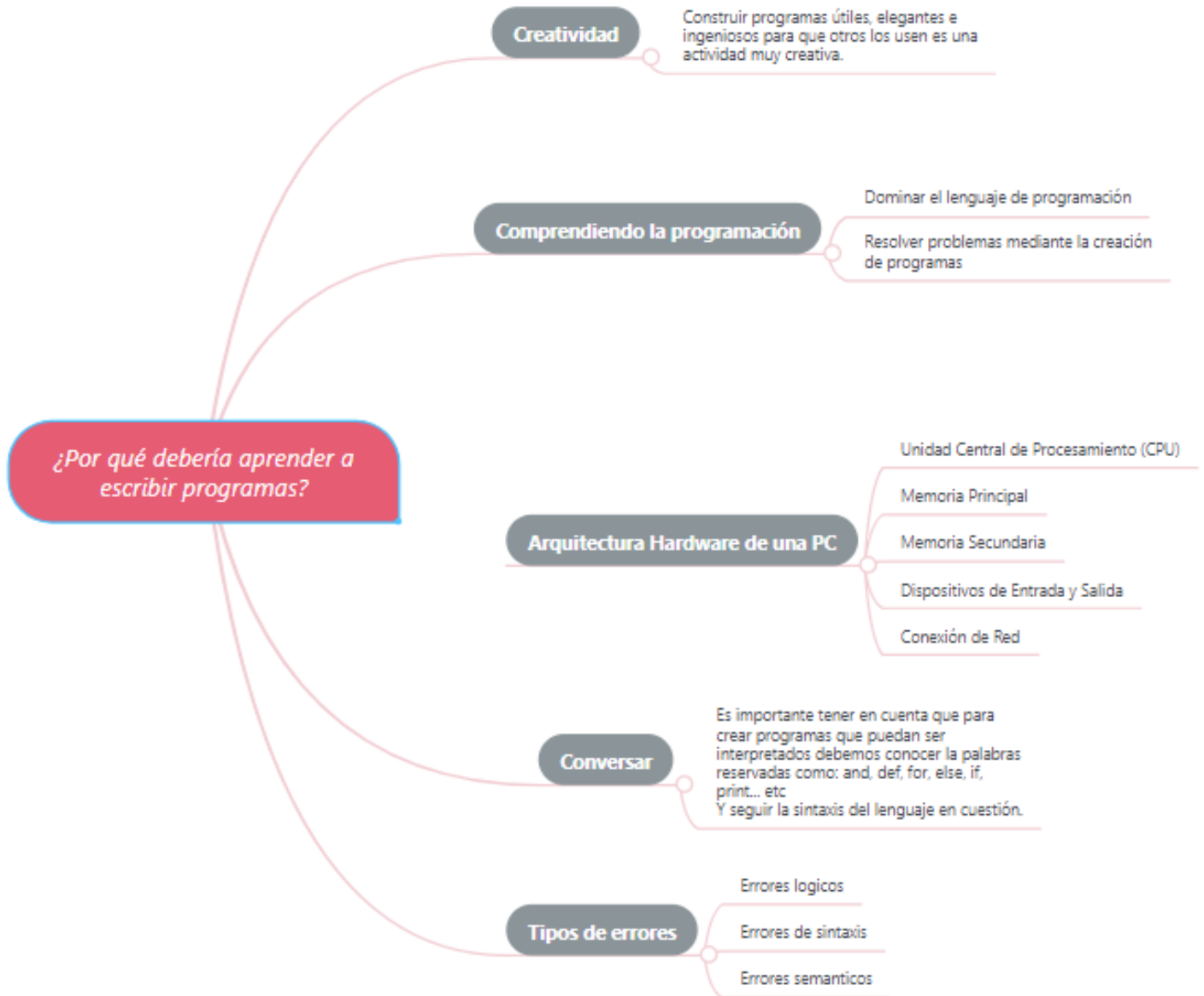
Grupo: 6

Fecha de entrega: martes 29 de agosto de 2017. (A más tardar a más 12 de la Noche).

- 1) Del libro **“Fundamentos de Programación”, Luis Joyanes Aguilar, et .al.** Que se adjunta con este documento, leer el capítulo 1, y hacer un cuadro sinóptico con los puntos más relevantes.



- 2) Del libro **Python para informáticos**, **Charles Severance**. Que se adjunta con este documento, leer el capítulo 1, y hacer un cuadro sinóptico con los puntos más relevantes. Y resolver los ejercicios de la sección 1.13.



Ejercicio 1.1 ¿Cuál es la función de la memoria secundaria en un PC?

- a) Ejecutar todos los cálculos y lógica del programa
- b) Recuperar páginas web de Internet
- c) Almacenar información durante mucho tiempo – incluso entre ciclos de apagado y encendido
- d) Recoger la entrada del usuario

Ejercicio 1.2 ¿Que es un programa?

Una secuencia de sentencias de Python, en este caso es el lenguaje que estamos usando, que han sido creadas para hacer algo.

Ejercicio 1.3 ¿Cuál es la diferencia entre un compilador y un intérprete?

Un intérprete lee el código fuente del programa tal y como lo ha escrito el programador, analiza ese código fuente e interpreta las instrucciones al vuelo.

Un compilador, en cambio, necesita que le entreguen el programa completo en un archivo, y después ejecuta un proceso para traducir el código fuente de alto nivel a código máquina. A continuación el compilador guarda el código máquina resultante en un archivo para su posterior ejecución

Ejercicio 1.4 ¿Cuál de los siguientes contiene “código máquina”?

- a) El intérprete de Python
- b) El teclado
- c) El código fuente de Python
- d) Un documento de un procesador de texto

Ejercicio 1.5 ¿Que está mal en el código siguiente?:

```
>>> print '¡Hola, mundo!'
File "<stdin>", line 1
print '¡Hola, mundo!'
^
SyntaxError: invalid syntax
>>>
```

El error está en que no es correcta la palabra “print”. La palabra correcta es “print”.

Ejercicio 1.6 ¿En qué parte del equipo queda almacenada una variable como “X” después de que se haya ejecutado la siguiente línea de Python?:

- ```
x = 123
```
- a) Unidad Central de Procesamiento

b) Memoria Principal

- c) Memoria Secundaria
- d) Dispositivos de Entrada
- e) Dispositivos de Salida

*Ejercicio 1.7 ¿Que imprimirá en pantalla el siguiente programa?:*

```
x = 43
x = x + 1
print x
```

a) 43

b) 44

- c)  $x + 1$
- d) Error, porque  $x = x + 1$  no es posible matemáticamente

*Ejercicio 1.8 Explica cada uno de los siguientes conceptos usando como ejemplo una capacidad humana:*

- (1) Unidad Central de Procesamiento – *El cerebro*
- (2) Memoria Principal – *Memoria a Corto Plazo*
- (3) Memoria Secundaria – *Memoria a Largo Plazo*
- (4) Dispositivo de Entrada - *Oídos*
- (5) Dispositivo de Salida - *Boca*

*Ejercicio 1.9 ¿Cómo puedes corregir un “Error de sintaxis”?*

Fijándonos primeramente en que línea se marca el error y a partir de ahí ver que regla de sintaxis hemos violado para corregirla.

3) Hacer los siguientes programas en lenguaje python, con su respectivo diagrama de flujo.

- Programa que resuelva ecuaciones de segundo grado, debe tomar en cuenta raíces reales y complejas, el usuario solo proporciona valores (a,b,c).
- Programa que calcule el valor del número (pi), a partir de la siguiente serie:

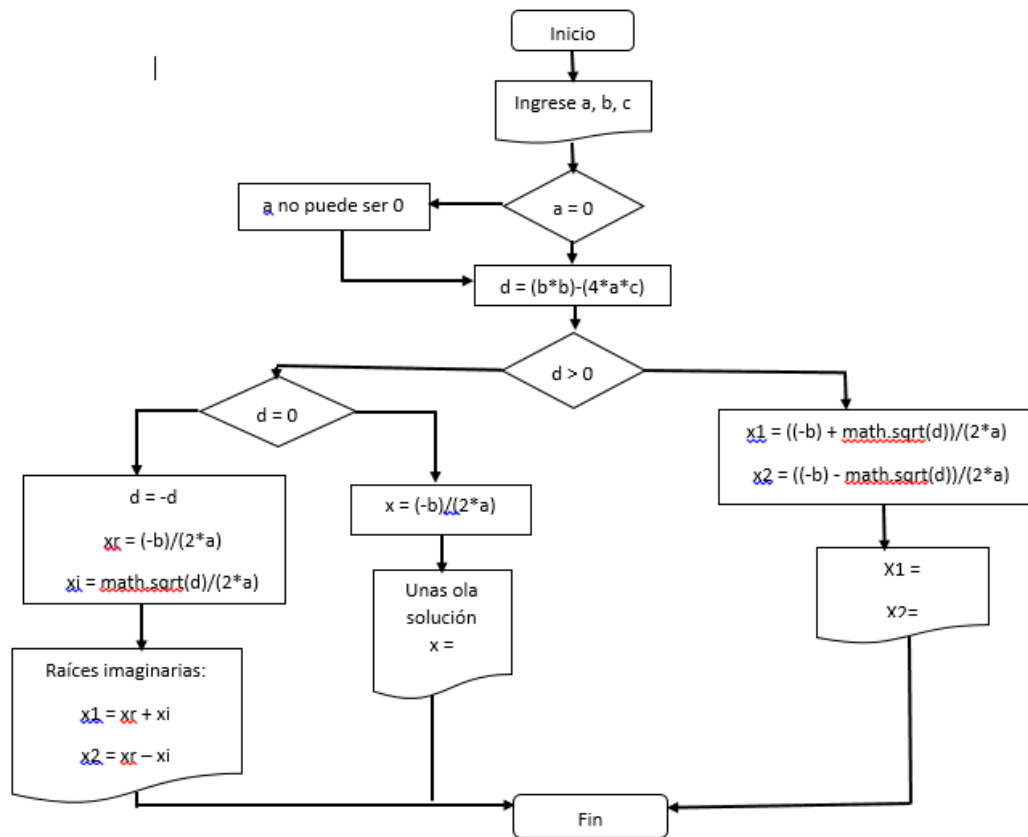
$$\pi = 4 - \frac{4}{3} + \frac{4}{5} - \frac{4}{7} + \frac{4}{9} - \frac{4}{11} + \dots$$

El usuario debe indicar cuantos términos debe llevar la serie.

- c) Programa que solicite al usuario un número entero positivo, y utilizando ciclos que imprima los siguientes patrones, por ejemplo N=10

[illegible]

a)



share
save
run

```

1 import math
2 print("Solución a una ecuación de 2do grado")
3 a = int(input("Ingrese a: "))
4 while a == 0:
5 a = int(input("a no puede ser 0, ingrese de nuevo: "))
6
7 b = int(input("Ingrese b: "))
8 c = int(input("Ingrese c: "))
9
10 d = ((b*b)-(4*a*c))
11
12 if d > 0 :
13 x1 = ((-b) + math.sqrt(d))/(2*a)
14 x2 = ((-b) - math.sqrt(d))/(2*a)
15 print("X1 = ",x1, "X2 = ", x2)
16
17 else:
18 if d == 0:
19 x = (-b)/(2*a)
20 print ("Solo existe una solución:")
21 print("x= ",x)
22 else:
23 d = -d
24 xr = (-b)/(2*a)
25 xi = math.sqrt(d)/(2*a)
26 print("Existen raíces imaginarias:")
27 print("x1 = ", xr, "+", xi, "i")
28 print("x2 = ", xr, "-", xi, "i")
29

```

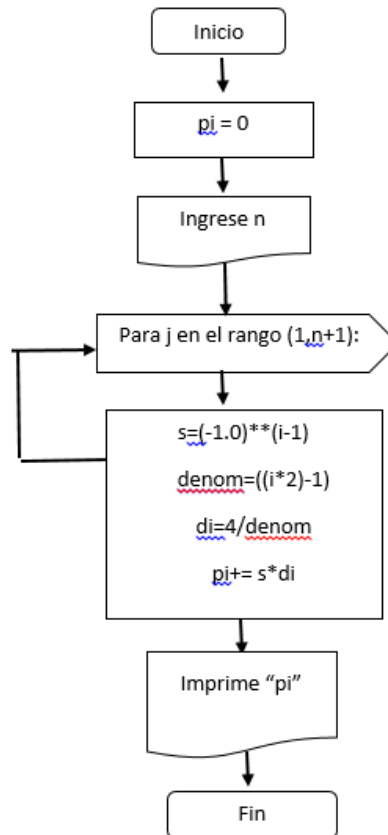
```

Python 3.6.1 (default, Dec 2015, 13:05:11)
[GCC 4.8.2] on linux
>
Solucion a una ecuación de 2do grado
Ingrese a: 1
Ingrese b: 2
Ingrese c: 1
Solo existe una solución:
x= -1.0

```

Programa1.Ecuacion2doGrado

b)

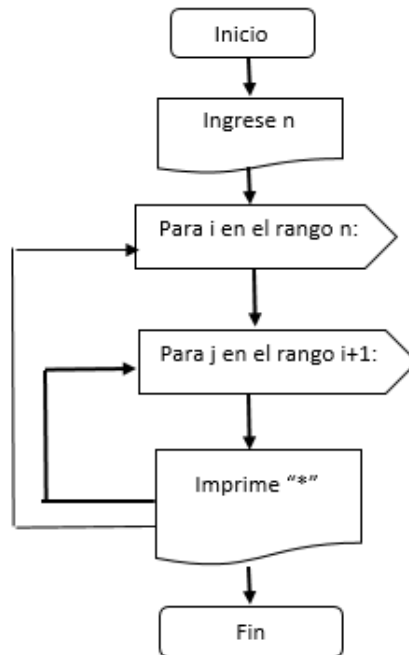


```
1 n=int(input("Introduce la cantidad de terminos: "))
2 pi=0
3 for i in range(1,n+1):
4 s=(-1.0)**(i-1)
5 denom=((i*2)-1)
6 di=4/denom
7 pi+= s*di
8 print("Pi =", pi)
9
```

Python 3.6.1 (default, Dec 2015, 13:05:11)  
[GCC 4.8.2] on linux  
Introduce la cantidad de terminos: 5  
Pi = 3.3396825396825403

Programa2.CalcularPi

c)



The screenshot shows the CodeSculptor interface. The left pane contains the following Python code:

```
1 n= int(input("Introduce un numero: "))
2 for i in range(n):
3 for j in range(i+1):
4 print("*"),
5 print("\n")
```

The right pane displays the output of the code, which is a pyramid of asterisks:

```
*
* *
* * *
* * * *
* * * * *
* * * * *
* * * * *
* * * * *
* * * * *
* * * * *
```

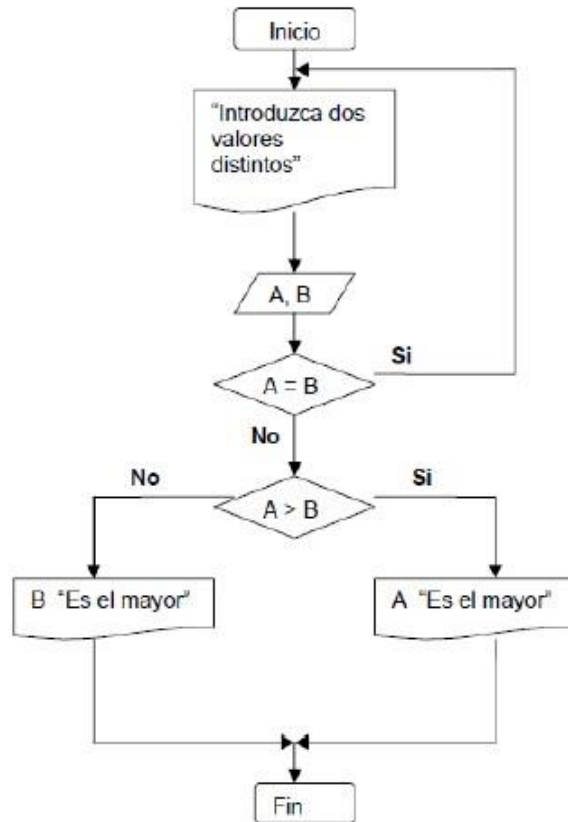
Programa3.1.Piramide





4) Programar los diagramas de Flujo siguientes:

a)



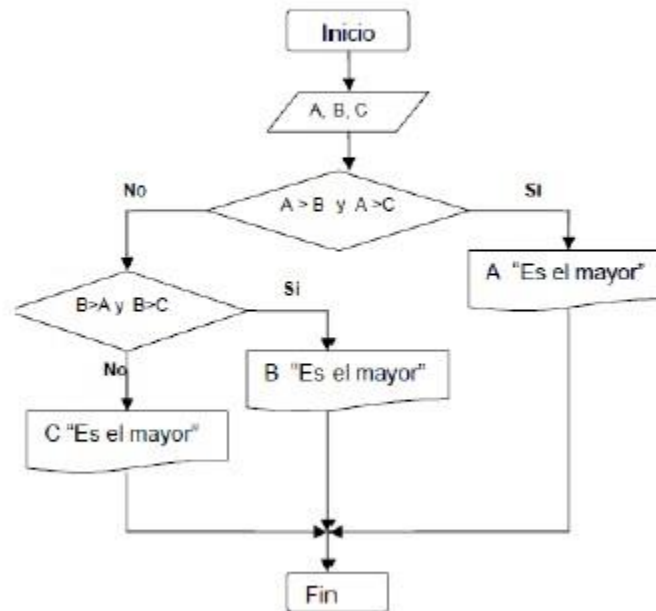
The screenshot shows the CodeSkulptor interface. On the left, the Python code is as follows:

```
1 print("Introduce dos numeros distintos")
2 a= int(input("Primer numero: "))
3 b= int(input("Segundo numero: "))
4
5 while a == b:
6 print("Son iguales, vuelve a intentar")
7 a= int(input("Primer numero: "))
8 b= int(input("Segundo numero: "))
9
10 if a > b:
11 print("El numero mayor es:",a)
12
13 else:
14 print("El numero mayor es:",b)
15
16
17
```

On the right, the output window shows the text: 'Introduce dos numeros distintos' followed by a new line and 'El numero mayor es:', 7)'. The interface includes a toolbar with icons for running, saving, and other functions, and a top bar with 'CodeSkulptor' branding and links to Docs, Demos, and Viz Mode.

Programa4.1.NumMayorDeDosNum

b)



```
1 a= int(input("Primer numero: "))
2 b= int(input("Segundo numero: "))
3 c= int(input("Tercer numero: "))
4
5
6
7 if a > b and a > c:
8 print("El numero mayor es:",a)
9
10 elif b > a and b > c:
11 print("El numero mayor es:",b)
12
13 else:
14 print("El numero mayor es:",c)
15
16
17
```

The screenshot shows the CodeSkulptor interface. On the left is the code editor with the Python code for finding the maximum of three numbers. On the right is the output window, which displays the result: `('El numero mayor es:', 9)`. The interface includes a toolbar at the top with icons for running, saving, and other functions, and tabs for 'Docs', 'Demos', and 'Viz Mode'.

Programa4.2.NumMayorDeTresNum

- 5) Hacer las modificaciones necesarias al código del programa de ordenamiento por método de la burbuja para que ordene un vector de caracteres alfabéticos, y un vector de cadenas.

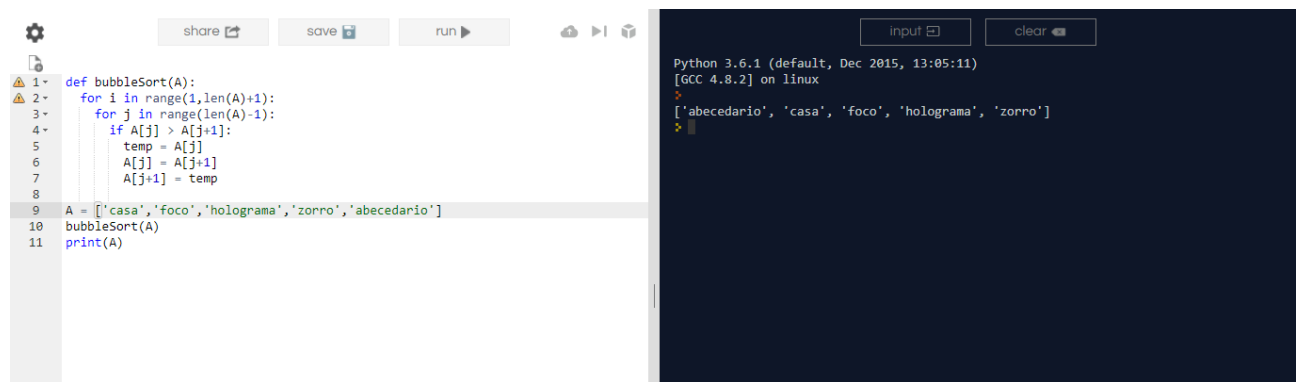


The screenshot shows a Python IDE with a code editor on the left and a terminal on the right. The code in the editor defines a bubble sort function and applies it to a list of characters. The terminal shows the output of the program.

```
def bubbleSort(A):
 for i in range(1, len(A)+1):
 for j in range(len(A)-1):
 if A[j] > A[j+1]:
 temp = A[j]
 A[j] = A[j+1]
 A[j+1] = temp
A = ['c', 'f', 'h', 'z', 'a']
bubbleSort(A)
print(A)
```

```
Python 3.6.1 (default, Dec 2015, 13:05:11)
[GCC 4.8.2] on linux
['a', 'c', 'f', 'h', 'z']
```

Programa5.1.BubbleSortCaracteres



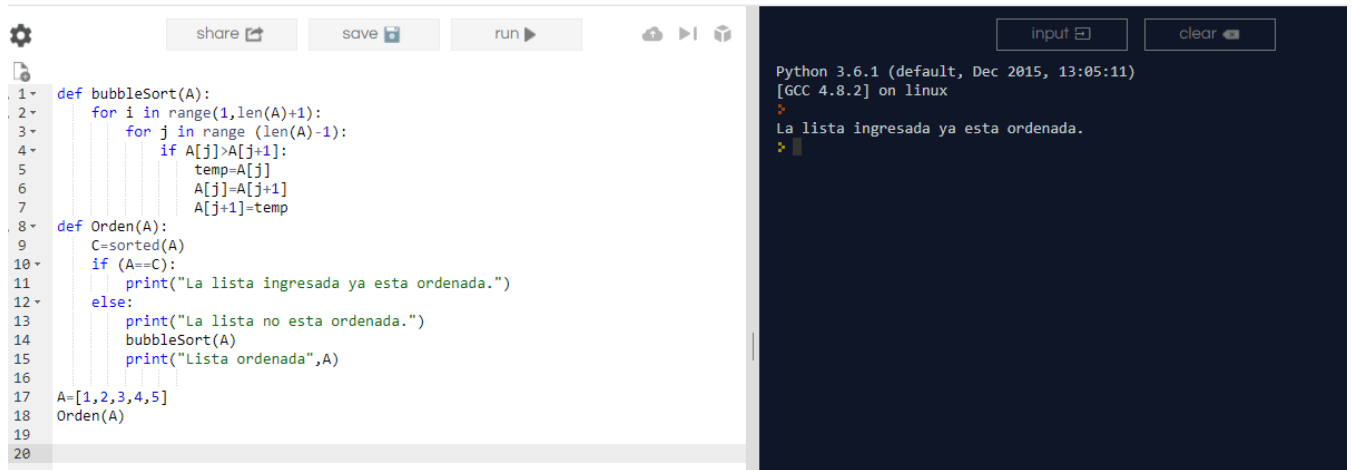
The screenshot shows a Python IDE with a code editor on the left and a terminal on the right. The code in the editor defines a bubble sort function and applies it to a list of strings. The terminal shows the output of the program.

```
def bubbleSort(A):
 for i in range(1, len(A)+1):
 for j in range(len(A)-1):
 if A[j] > A[j+1]:
 temp = A[j]
 A[j] = A[j+1]
 A[j+1] = temp
A = ['casa', 'foco', 'holograma', 'zorro', 'abecedario']
bubbleSort(A)
print(A)
```

```
Python 3.6.1 (default, Dec 2015, 13:05:11)
[GCC 4.8.2] on linux
['abecedario', 'casa', 'foco', 'holograma', 'zorro']
```

Programa5.2.BubbleSortCadena

- 6) Hacer las modificaciones necesarias al código del programa de ordenamiento por método de la burbuja para que si se ingresa un vector con enteros, y si se encuentran ya ordenados, no se entre al proceso de las comparaciones.



```
def bubbleSort(A):
 for i in range(1, len(A)+1):
 for j in range (len(A)-1):
 if A[j]>A[j+1]:
 temp=A[j]
 A[j]=A[j+1]
 A[j+1]=temp
def Orden(A):
 C=sorted(A)
 if (A==C):
 print("La lista ingresada ya esta ordenada.")
 else:
 print("La lista no esta ordenada.")
 bubbleSort(A)
 print("Lista ordenada",A)
A=[1,2,3,4,5]
Orden(A)
```

Python 3.6.1 (default, Dec 2015, 13:05:11)  
[GCC 4.8.2] on linux  
La lista ingresada ya esta ordenada.

Programa6.BubbleSortModificado

- 7) Leer la presentación proporcionada por el profesor.

Nota: Para todos los programas que se realicen, en el reporte se deberá mostrar evidencia de que el código funciona (Pantalla de código y corrida del código). Y se deberá adjuntar en un ZIP el código de los mismos.

Programas Anexos en Archivo ZIP "grupo6\_EDA\_Programas".