

**PROYECTO 1**

**RELACIONES Y SUS PROPIEDADES**

**Matemáticas Computacionales**  
**Dr. Victor de la Cueva**

**Hecho por:**  
**Rafael Correa A01019498**

**06/06/2016**

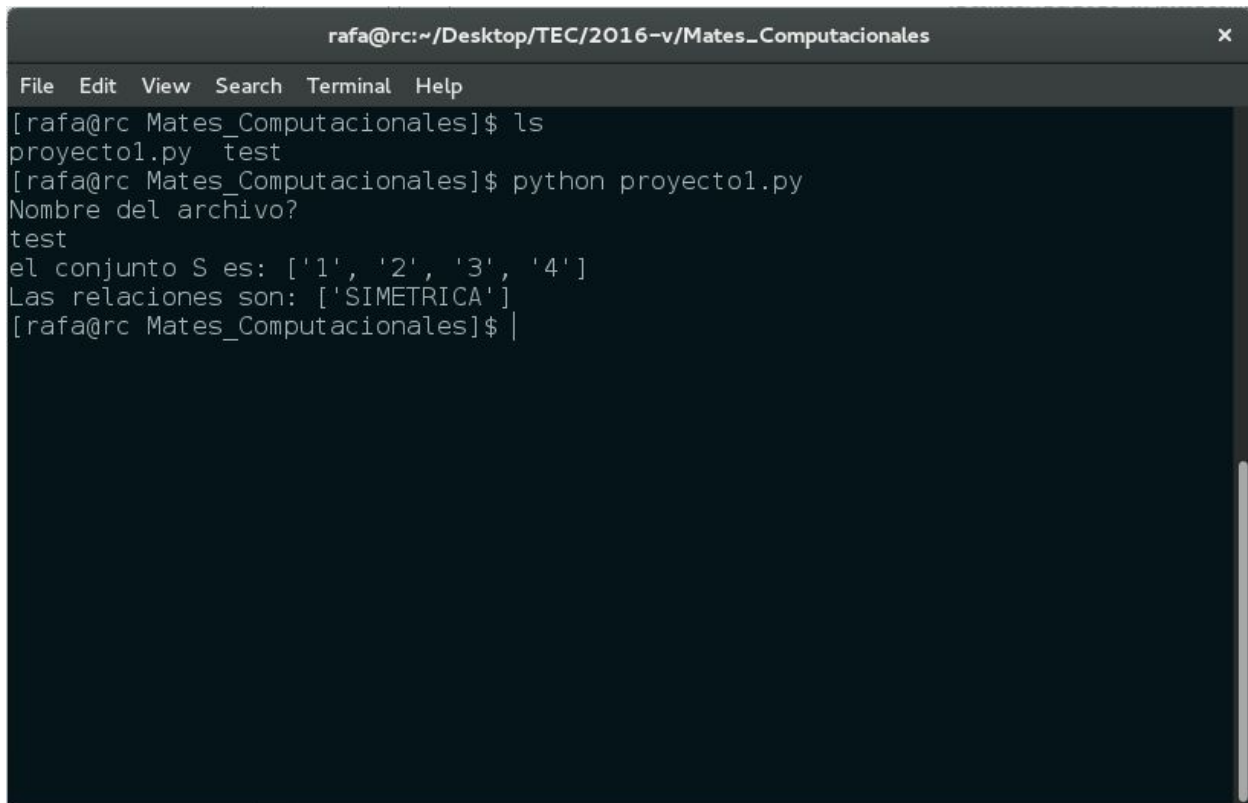
## Manual de Usuario

Crear un archivo con el formato especificado (la primera línea debe indicar cuántos pares existen, las demás deben especificar los pares).

```
1 5
2 1 2
3 2 1
4 3 4
5 4 3
6 1 1|
```

Plain Text ▼ Tab Width: 4 ▼ Ln 6, Col 4 ▼ INS

Correr el script de python (con python2.7) y escribir el nombre del archivo:



The screenshot shows a terminal window titled 'rafa@rc:~/Desktop/TEC/2016-v/Mates\_Computacionales'. The terminal has a menu bar with 'File', 'Edit', 'View', 'Search', 'Terminal', and 'Help'. The command prompt shows the user running 'ls', which lists 'proyecto1.py' and 'test'. Then, the user runs 'python proyecto1.py', which prompts for the file name. The user enters 'test', and the script outputs: 'el conjunto S es: ['1', '2', '3', '4']' and 'Las relaciones son: ['SIMETRICA']'. The prompt returns to the user.

```
rafa@rc:~/Desktop/TEC/2016-v/Mates_Computacionales
File Edit View Search Terminal Help
[rafa@rc Mates_Computacionales]$ ls
proyecto1.py  test
[rafa@rc Mates_Computacionales]$ python proyecto1.py
Nombre del archivo?
test
el conjunto S es: ['1', '2', '3', '4']
Las relaciones son: ['SIMETRICA']
[rafa@rc Mates_Computacionales]$ |
```

## Código Fuente

```
#funcion que regresa dos listas: una es el set de los elementos, la otra una lista de tuplas
con las relaciones
def parseFile(archivo):
    f = open(archivo, 'r')
    cuantos = int(f.readline())
    set1 = []
    relaciones = []
    for line in f:
        par = line.split()
        relaciones.append(par)
        if par[0] not in set1:
            set1.append(par[0])
        if par[1] not in set1:
            set1.append(par[1])
    return set1, relaciones
```

#funcion que al recibir las listas del set y de las relaciones, determina si se es reflexiva

```
def testReflexiva(set1, relaciones):  
    for elemento in set1:  
        test = [elemento, elemento]  
        if test not in relaciones:  
            return False  
  
    return True
```

#funcion que al recibir las listas del set y de las relaciones, determina si se es irreflexiva

```
def testIrreflexiva(set1, relaciones):  
    for elemento in set1:  
        test = [elemento, elemento]  
        if test in relaciones:  
            return False  
  
    return True
```

#funcion que al recibir las listas del set y de las relaciones, determina si se es transitiva

```
def testTransitiva(set1, relaciones):  
    for relacion in relaciones:  
        a = relacion[0]  
        b = relacion[1]  
        for busqueda in relaciones:  
            if busqueda[0] == b:  
                c = busqueda[1]  
                if([a,c] not in relaciones):  
                    return False  
  
    return True
```

#funcion que al recibir las listas del set y de las relaciones, determina si se es simetrica

```
def testSimetria(set1, relaciones):  
    for relacion in relaciones:  
        a = relacion[0]  
        b = relacion[1]  
        if([b,a] not in relaciones):  
            return False  
  
    return True
```

#funcion que al recibir las listas del set y de las relaciones, determina si se es asimetrica

```
def testAsimetria(set1, relaciones):  
    for relacion in relaciones:  
        a = relacion[0]  
        b = relacion[1]  
        if([b,a] in relaciones):  
            return False  
  
    return True
```

#funcion que al recibir las listas del set determina las relaciones con las que se cumple

```
def testAll(set1, relaciones):  
    respuesta = []
```

```

if(testReflexiva(set1, relaciones)):
    Respuesta.append('REFLEXIVA')
if(testIrreflexiva(set1, relaciones)):
    respuesta.append('IRREFLEXIVA')
if(testTransitiva(set1, relaciones)):
    respuesta.append('TRANSITIVA')
if(testSimetria(set1, relaciones)):
    respuesta.append('SIMETRICA')
if(testAsimetria(set1, relaciones)):
    respuesta.append('ASIMETRICA')

if not len(respuesta):
    respuesta.append('NO CUMPLE NINGUNA')

return respuesta

#preguntar el nombre del archivo
archivo = raw_input("Nombre del archivo?\n")

#llamar la funcion con el nombre del archivo
set1, relaciones = parseFile(archivo)

#resultados
print 'El conjunto S es: %s' % set1
print 'Las relaciones son: %s' % testAll(set1, relaciones)

```

## Documentación

### FUNCIONES

- **parseFile(archivo)**
  - Recibe como parámetro un string con el nombre del archivo.
  - Parsea el archivo y guarda la información en dos listas que regresa al final.
- **testPropiedad(set1, relaciones)**
  - Existen 5 funciones de estas: testReflexiva(), testIrreflexiva(), testTransitiva(), testSimetria() y testAsimetria()
  - Todas las funciones reciben como parámetro dos listas: una con los elementos del set y otra con las tuplas de las relaciones.
  - Todas las funciones regresan True o False para la propiedad que prueban.
  - Las funciones funcionan de la siguiente manera:

- Para probar la reflexividad, para cada elemento  $a$  en  $S$ , se busca la tupla  $(a, a)$  en la lista de relaciones. Si no se cumple con esta condición, se termina el procedimiento y regresa Falso. De manera similar para la irreflexividad, se busca la tupla  $(a,a)$  y si se encuentra, regresa Falso.
- Para probar la transitividad, para cada relación  $(a,b)$  en la lista de relaciones, se toma el segundo elemento y se busca cualquier tupla  $(b,c)$ . Se busca en la lista la tupla  $(a,c)$ , si no se encuentra, regresa Falso.
- Para probar la simetría, para cada relación  $(a,b)$  en la lista de relaciones, se busca la tupla  $(b,a)$ . Si no se encuentra regresa Falso. De manera similar para la asimetría, se busca la tupla  $(b,a)$  y si se encuentra regresa Falso.
- **testAll(set1, relaciones)**
  - Esta función recibe los mismos parámetros que **testPropiedad(set1, relaciones)**.
  - Llama las 5 funciones mencionadas anteriormente (las de **testPropiedad(set1, relaciones)**).