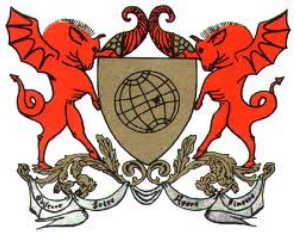
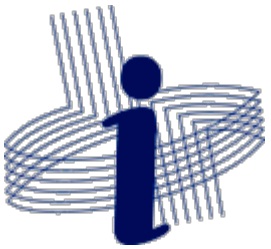


Universidade Federal de Viçosa
Departamento de Informática
Centro de Ciências Exatas e Tecnológicas



INF 100 – Introduction to Programming

Arrays

Motivation

What does the program below do?

```
while True:
    n = int(input('How many values will be typed? '))
    if n > 0:
        break
    print('Must be greater than zero')

sum = 0
for i in range(0, n):
    v = float(input('Type a value: '))
    sum += v

print('Average = ', sum/n)
```


Motivation

The program below calculates the average of a sequence of values.

```
while True:
    n = int(input('How many values will be typed? '))
    if n > 0:
        break
    print('Must be greater than zero')

sum = 0
for i in range(0, n):
    v = float(input('Type a value: '))
    sum += v

print('Average = ', sum/n)
```


Motivation

How can we calculate how many values are greater than the average?

```
while True:
    n = int(input('How many values will be typed? '))
    if n > 0:
        break
    print('Must be greater than zero')

sum = 0
for i in range(0, n):
    v = float(input('Type a value: '))
    sum += v

print('Average = ', sum/n)
```


Motivation

```
while True:
    n = int(input('How many values will be typed? '))
    if n > 0:
        break
    print('Must be greater than zero')

sum = 0
for i in range(0, n):
    v = float(input('Type a value: '))
    sum += v

print('Average = ', sum/n)
```

All the typed values must be stored...

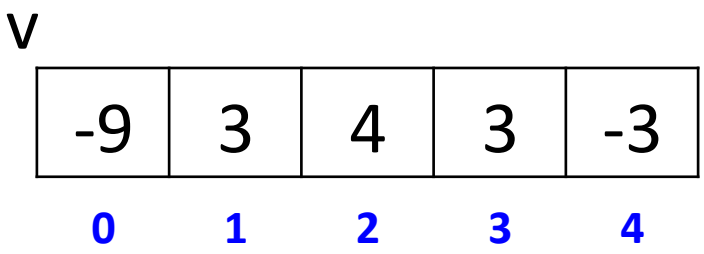
... so that they can later be compared with the calculated average

Arrays

- Arrays are *aggregates* (set of variables) in which the elements are indexed by integer numbers.
- The index of the first element is always 0 (zero).
- An array can have one, two, or several dimensions.

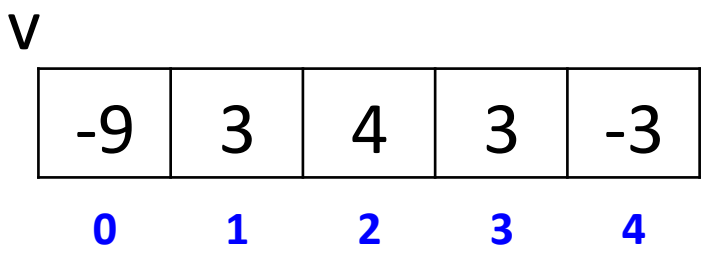

Arrays

- An array can be represented as a sequence of cells storing values and identified by an integer number.
- For instance, an array v containing 5 integer numbers can be represented as:



Arrays

- In this example, the element stored in the 5th position can be accessed by v[4] and this element is the value -3. v[0] is -9, v[1] is 3 and so on.



- Note that two different cells can store a same value (for instance, v[1] and v[3] store the value 3).

Arrays in Python

- In order to create and manipulate arrays of numerical values in Pyhton, we are going to use the library **`numpy`**.
- Advantages of using numpy:
  - Easy way for creating and initializing arrays.
  - The syntax is similar to other programming languages.


Arrays in Python

- First, it is necessary import the library:

```
import numpy
```

- When importing a library, it may be convenient to give and alias that is shorter than the name of the library. In this case, the alias can replace the name of the library in the program.

```
import numpy as np
```


Arrays in Python

- Creating an array with *n* real numbers, not initialized:

```
identifier = numpy.empty( n )
```

where **identifier** is the name for the array.

- Example:

```
import numpy

grades = numpy.empty( 5 )
heights = numpy.empty( 10 )
```


Arrays in Python

- Using the alias **np** for **numpy**, the program would be:

```
import numpy as np

grades = np.empty( 5 )
heights = np.empty( 10 )
```


Arrays in Python

- Creating arrays with *n* real numbers, and initializing with zeros:

```
import numpy as np

grades = numpy.zeros( 5 )
heigths = numpy.zeros( 10 )
```


Arrays in Python

```
v = np.empty( 5 )  # Creates array with 5 elements

v[0] = 2       # the value 2 is stored in the first position
v[1] = 4.5     # 4.5 is stored in the second position
v[2] = -1      # -1 is stored in the third position
v[3] = 0.5     # 0.5 is stored in the fourth position
v[4] = 4.5     # 4.5 is stored in the fifth position

print( v[4] + v[1] )   # prints 9.0 (=4.5+4.5) on the screen
v[0] = v[2] = -10      # -10 is stored in the 3rd and 1st positions
v[3] = float( input('...'))
# reads a  value from the keyboard and stores it in v[3]

v[4] = v[3] + v[2]*v[2]   # accessing several cells in a command

# Error: accessing invalid index
print( v[-1], v[5], v[1.5] )
```

Be careful!

Arrays in Python

- Initializing arrays:

```
A = np.asarray([4, 3, 2, 1])

grades = np.asarray([70.5, 80, 54.3, 77.8])
```


Accessing the cells of an array

- When an algorithm uses arrays, it is frequently necessary to access all the stored elements and execute an operation on each element. Examples:

```
for each position i of the array A:
   print A[i] on the screen
```

```
for each position i of the array A:
   multiply A[i] by 2
```


Accessing the cells of an array

- Translation to Python using the command **`while`**:

```
for each position i of the array A:
    print A[i] on the screen
```

```
i = 0
while i < len( A ):
    print( A[i] )
    i = i + 1
```

Accessing the cells of an array

- Using the command **for** :

```
for i = 0 ... n-1:
   Print A[i]
(where n = size of A)
```

(algorithm)

```
for i in range(0, len( A )):
    print( A[i] )
```

(Python)

Accessing the cells of an array

- The command **for** has an alternative version that may be more convenient in some situations:

```
for each x ∈ A:
    print x
```

(algorithm)

```
for x in A:
    print( x )
```

(Python)


Accessing the cells of an array (more examples)

```
import numpy as np

n = int( input('Type the number of elements: '))
A = np.empty( n )  # creates an array

# Reading values from keyboard and storing them in the array
for i in range(0, n):
    s = 'Type element of position %d:' % (i+1)
    A[i] = float( input( s ))

# Multiply all values by 2
for i in range(0, n):
    A[i] = A[i] * 2

# Print values on the screen
for x in A:
    print( x )
```


Accessing the cells of an array (more examples)

```
import numpy as np

n = int( input('Type the number of elements: '))
A = np.zeros( n )  # creates an array with values 0.0

# Prints the array on the screen, with elements separated by
blank for x in A:
    print( x, end=' ')
print()

# Prints the array on the screen, with elements separated by
tab for x in A:
    print( x, end='\t')
print()

# Prints the array on the screen, with formatting
for x in A:
    print('%7.3f' % (x), end='')
print()
```

```
import numpy as np

while True:
    n = int(input('How many values will be typed? '))
    if n > 0:
        break
    print('Must be greater than zero')

A = np.zeros(n)
sum = 0
for i in range(0, n):
    v = float(input('Type a value: '))
    A[i] = v
    sum += v

print('Typed values:', end='')
for x in A:
    print('%7.2f' % (x), end='')
average = sum/n
print('\nAverage = ', average)

g = 0
for x in A:
    if x > average:
        g += 1
print('Number of values greater than the average =', g)
```

Python program that calculates the average and the number of values greater than the average

Exercise #1

- Write a Python program that reads the grades of n students and then calculates and prints:
  a) The average of the grades: $$\overline{x} = \frac{\sum_{i=1}^{n} x_i}{n}$$
  b) The standard deviation of the grades:
  $$s = \sqrt{\frac{\sum_{i=1}^{n} (x_i - \overline{x})^2}{n-1}}$$
  c) The largest grade
  d) The smallest grade
  e) The number of grades smaller than the average


Solution – part 1

```
import numpy as np

while True:
    n = int(input('How many values will be typed? '))
    if n > 0:
        break
    print('Must be greater than zero')

A = np.zeros(n)
sum = 0
for i in range(0, n):
    v = float(input('Type a value: '))
    A[i] = v
    sum += v

print('Typed values:', end='')
for x in A:
    print('%7.2f' % (x), end='')
average = sum/n
print('\nAverage = ', average)
```


Solution – part 2

```
sum2 = 0
for x in A:
    sum2 += (x - average)**2
sdev = (sum2 / (n-1))**0.5
print('Standard deviation = ', sdev)

smallest = largest = A[0]
for i in range (1,n):
    if A[i] < smallest:
        smallest = A[i]
    elif A[i] > largest:
        largest = A[i]
print('Largest value =', largest)
print('Samllest value =', smallest)

k = 0
for x in A:
    if x < average:
        k += 1
print('Number of values smaller than the average =', k)
```


Exercise #2

Write a Python program that:

a) Read a sequence of integer values.

b) Prints on the screen the relation between the subsequent values.

Example:

```
Type the number of values: 5
Type the values (one value per line):
0
4
-1
-1
5
Relations: 0 < 4 > -1 = -1 < 5
```

Solution

```
import numpy as np

while True:
    n = int(input('How many values will be typed? '))
    if n > 0:
        break
    print('Must be greater than zero')

A = np.zeros(n)
print('Type the values (one value per line):')
for i in range(0, n):
    A[i] = float(input(''))

print(A[0], end=' ')
for i in range (1,n):
    if A[i-1] < A[i]:
        print('<', end=' ')
    elif A[i-1] > A[i]:
        print('>', end=' ')
    else:
        print('=', end=' ')
    print(A[i], end=' ')
```