ITD62-123 COMPUTER PROGRAMMING

IMI62-122 FUNDAMENTAL OF COMPUTER PROGRAMMING

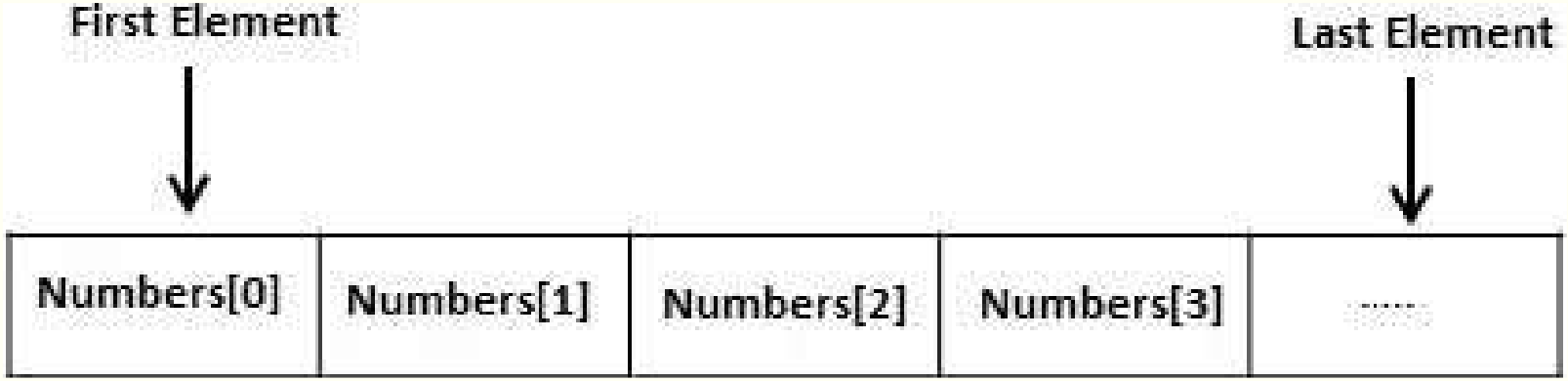
Theerat Saichoo & Yunyong Punsawad
School of Informatics, Walailak University

Chapter 5
Arrays and String

Theerat Saichoo
School of Informatics, Walailak University

Arrays

- Arrays a kind of data structure that can store a **fixed-size sequential** collection of elements of the **same type.**
- An array is used to store a collection of data, but it is often more useful to think of an array as a collection of variables of the same type.



Python Arrays

- Python does not have built-in support for Arrays, but Python Lists can be used instead.
- To work with arrays in Python, We will have to import the NumPy library.
- NumPy is a Python library used for working with arrays.
- It also has functions for working in domain of linear algebra, fourier transform, and matrices.
- NumPy was created in 2005 by Travis Oliphant. It is an open source project and you can use it freely.

Python Arrays

- **Installation of NumPy**

```
pip install numpy
```

- **Import NumPy**

```
import numpy              #normal import
import numpy as np        #import with alias
```

NumPy Creating Arrays

- NumPy is used to work with arrays. The array object in NumPy is called ndarray.
- **Example**

```
import numpy as np

arr = np.array([1, 2, 3, 4, 5])
```

2-D Arrays

- An array that has 1-D arrays as its elements is called a 2-D array.
- These are often used to represent matrix or 2nd order tensors.
- Create a 2-D array containing two arrays with the values 1,2,3 and 4,5,6:
- Example

```
arr = np.array([[1, 2, 3], [4, 5, 6]])
```

NumPy Array Indexing

- Array indexing is the same as accessing an array element.
- You can access an array element by referring to its index number.
- The indexes in NumPy arrays start with 0, meaning that the first element has index 0, and the second has index 1 etc.
- **Example**

```
import numpy as np
arr = np.array([1, 2, 3, 4])
print(arr[2] + arr[3])
```

Access 2-D Arrays

- To access elements from 2-D arrays we can use comma separated integers representing the dimension and the index of the element.
- **Example 1 -** Access the 2nd element on 1st dim:

```
import numpy as np
arr = np.array([[1,2,3,4,5], [6,7,8,9,10]])
print('2nd element on 1st dim: ', arr[0, 1])
```

Access 2-D Arrays

- **Example 2** - Access the 5th element on 2nd dim:

```
import numpy as np
arr = np.array([[1,2,3,4,5], [6,7,8,9,10]])
print('5th element on 2nd dim: ', arr[1, 4])
```

NumPy Data Types

- NumPy has some extra data types, and refer to data types with one character, like i for integers, u for unsigned integers etc.
- Below is a list of all data types in NumPy and the characters used to represent them.

I - integer
b - boolean
u - unsigned integer
f - float
c - complex float
m - timedelta
M - datetime

O - object
S - string
U - unicode string
V - fixed chunk of memory for other type (void)

Checking the Data Type of an Array

- The NumPy array object has a property called dtype that returns the data type of the array.
- **Example**

```
import numpy as np

arr = np.array([1, 2, 3, 4])

print(arr.dtype)
```

Creating Arrays With a Defined Data Type

- **Example 1 -** Create an array with data type string:

```
import numpy as np

arr = np.array([1, 2, 3, 4], dtype='S')

print(arr)
print(arr.dtype)
```

Creating Arrays With a Defined Data Type

- **Example 2 -** Create an array with data type 4 bytes integer:

```
import numpy as np

arr = np.array([1, 2, 3, 4], dtype='i4')

print(arr)
print(arr.dtype)
```

Converting Data Type on Existing Arrays

- The best way to change the data type of an existing array, is to make a copy of the array with the astype() method.
- The astype() function creates a copy of the array, and allows you to specify the data type as a parameter.
- The data type can be specified using a string, like 'f' for float, 'i' for integer etc. or you can use the data type directly like float for float and int for integer.

Converting Data Type on Existing Arrays

- **Example -** Change data type from float to integer by using int as parameter value:

```
import numpy as np

arr = np.array([1.1, 2.1, 3.1])

newarr = arr.astype(int)

print(newarr)
print(newarr.dtype)
```

Iterating Arrays

- Iterating means going through elements one by one.
- As we deal with multi-dimensional arrays in numpy, we can do this using basic for loop of python.
- If we iterate on a 1-D array it will go through each element one by one.
- In a 2-D array it will go through all the rows.

Iterating Arrays

- **Example**

```
import numpy as np

arr = np.array([1, 2, 3])

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

Iterating 2-D Arrays

- **Example -** Iterate on the elements of the following 2-D array:

```
import numpy as np

arr = np.array([[1, 2, 3], [4, 5, 6]])

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

Iterating 2-D Arrays

- **Example -** iterate on each scalar element of the 2-D array:

```
import numpy as np

arr = np.array([[1, 2, 3], [4, 5, 6]])

for x in arr:
        for y in x:
                print(y)
```

Searching Arrays

- We can search an array for a certain value, and return the indexes that get a match.
- To search an array, use the where() method.
- **Example -** Find the indexes where the value is 4:

```
import numpy as np
arr = np.array([1, 2, 3, 4, 5, 4, 4])
x = np.where(arr == 4)
print(x)
```

Python Strings

- Strings in python are surrounded by either single quotation marks, or double quotation marks.
- 'hello' is the same as "hello".
- Strings are Arrays.
- Square brackets can be used to access elements of the string.
- **Example**

```
a = "Hello, World!"
print(a[1])
```

Looping Through a String

- Since strings are arrays, we can loop through the characters in a string, with a for loop.
- **Example -** Loop through the letters in the word "banana":

```
for x in "banana":
    print(x)
```

Check String

- To check if a certain phrase or character is present in a string, we can use the keyword in.
- **Example -** Check if "free" is present in the following text:

```
txt = "The best things in life are free!"

print("free" in txt)
```

Check String

- Use it in an if statement:
- **Example** - Print only if "free" is present:

```
txt = "The best things in life are free!"

if "free" in txt:
        print("Yes, 'free' is present.")
```

Check if NOT

- To check if a certain phrase or character is NOT present in a string, we can use the keyword not in.
- **Example -** Check if "expensive" is NOT present in the following text:

```
txt = "The best things in life are free!“

print("expensive" not in txt)
```

Check if NOT

- Use it in an if statement:
- **Example** - print only if "expensive" is NOT present:

```
txt = "The best things in life are free!"

if "expensive" not in txt:
        print("Yes, 'expensive' is NOT present.")
```

Special thanks to W3Schools