Introduction to Programming

UNIVERSITY OF CAPE TOWN
IYUNIVESITHI YASEKAPA • UNIVERSITEIT VAN KAAPSTAD

Arrays

Basic Operations

CSC1

Hussein Suleman
Department of Computer Science

Basic Operations

Hussein Suleman
Department of Computer Science
University of Cape Town

Problem 1: Basic statistics program

Write a program to output the average (a), the median (x), and the standard deviation (s), of a series of test values typed in by the user.

- The median is the middle value in a list of values.
- The standard deviation is a measure of how spread-out the data is and is calculated using this formula:

$$s = \sqrt{\frac{\sum(\overline{x} - x_i)^2}{n-1}}$$

standard deviation: a quantity expressed by how much the members of a group differ from the mean value for the group.

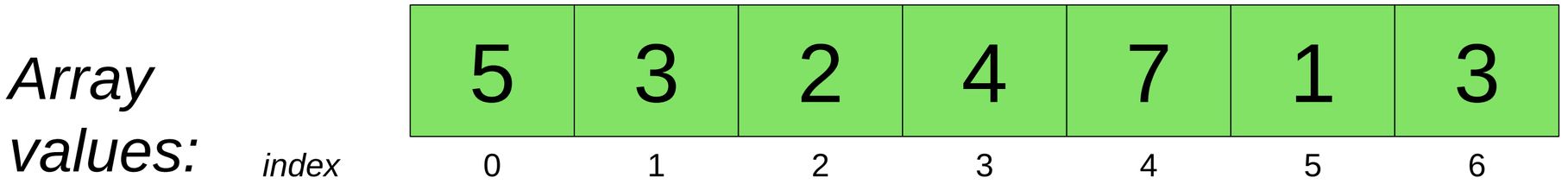
Problem 1: Basic statistics program

- We can calculate the average.
  - arrays_problem1_soln.py

- … but we don’t yet have the tools for the median and standard deviation. We need to be able to store and sort items in a list somehow.

Concept: Arrays

An array is an **indexed sequence** of values associated with **one variable**.



- Arrays can be fixed length or variable length.
- Arrays can hold multiple values of the same type or different types.

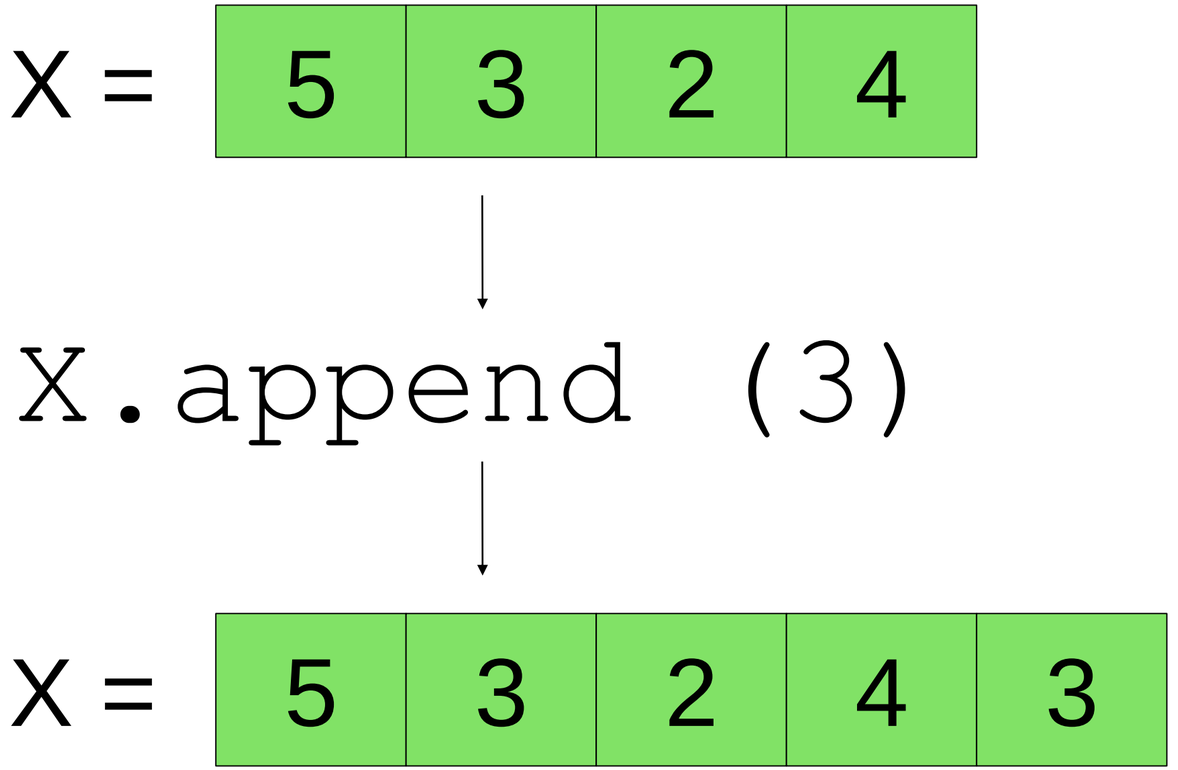
Python Arrays: Lists

- In Python, an array is called a list.
- To create a list:

```
list1 = [ ]          # empty list
nums = [ 1, 2, 3, 4, 7, 8, 9, 10, 13, 14, 15 ]
       # list of numbers
animals = ['cat', 'dog', ’baboon', 'bison’]
   # list of strings
stuff = [ 1, 2, 'hello' ]
       # mixed type list (yes we can do that)
```

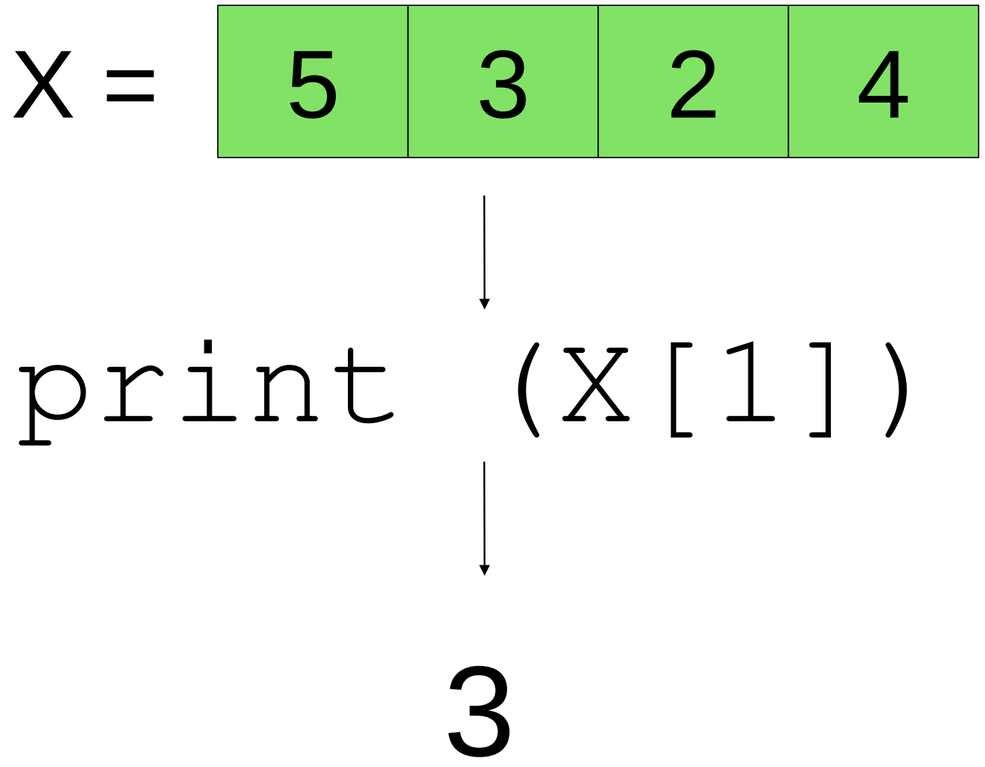
Common Operations 1/5

Adding an item to a list



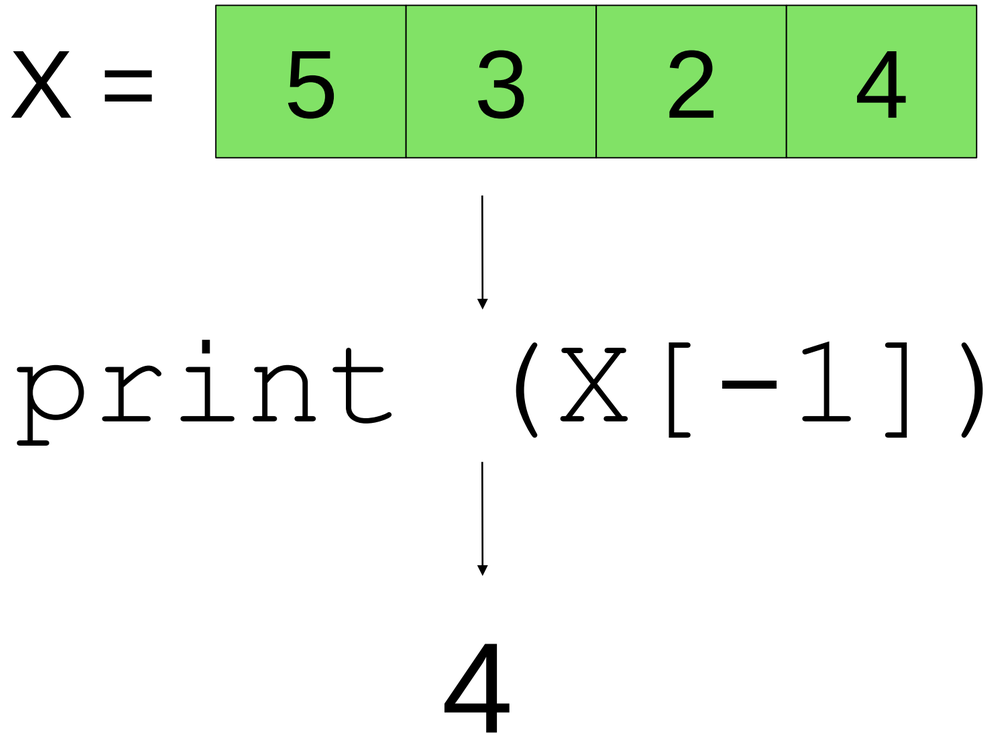
Common Operations 2/5

Accessing an item in a list



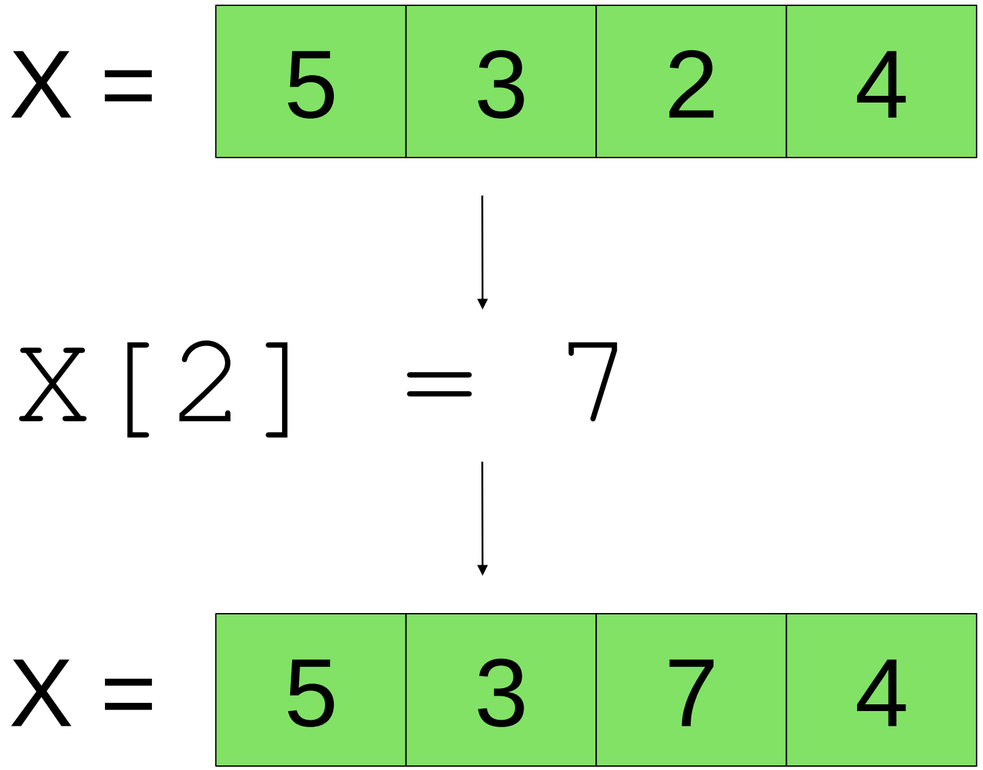
Common Operations 3/5

Accessing the last item in a list



Common Operations 4/5

Changing an item in a list



Common Operations 5/5

- Iterating over items in list – processing each item in the list individually.
- When you do **not** need to know the **indices**:

```
for a in X:
  print(a)
```

- When you do need to know the indices:

```
for n in range (len (X)):
  print(n, X[n])
```

Example

This function finds the first occurrence of an item in a list. If the item does not exist, the function returns -1.

```
def index (values, item):
    for i in range (len(values)):
        if values[i] == item:
            return i
    return -1
```

What values are needed to test this program using different testing strategies?

Exercise

- Write a program to take 5 strings as input from the user and store them in an array. Then print them out in the same order from the array.

UNIVERSITY OF CAPE TOWN

IYUNIVESITHI YASEKAPA • UNIVERSITEIT VAN KAAPSTAD

SCHOOL OF IT

Introduction to Programming

UNIVERSITY OF CAPE TOWN
IYUNIVESITHI YASEKAPA • UNIVERSITEIT VAN KAAPSTAD

Arrays

Using Array Operations

CSC1

Hussein Suleman
Department of Computer Science

Using Array Operations

Hussein Suleman
Department of Computer Science
University of Cape Town

Common Pitfall/Error

Accessing an item that is not in the list!

- Python's response:

```
Traceback (most recent call last):
  File "<string>", line 1, in <fragment>
builtins.IndexError: list index out of range
```

- Solution:
  - Check the list length first and make sure the item exists!

Basic List Manipulation

Operation	Syntax	Example	Example output
Merging lists (concatenation)	<list1> + <list2>	X = [1,2] Y = [3,4]	[1,2,3,4]
Checking for item (**membership**)	<item> in <list>	X = [1, 2] 1 in X	True
Multiplying content of lists (repetition)	<list> * <n>	X = [1,2] X * 2	[1,2,1,2]
Access list item **(indexing)**	<list>[<index>]	X = [1,2,3] X[2]	3
Get length of list **(length)**	len(<list>)	X = [1,2,3,4] len(X)	4
Delete item	del <list>[<i>]	X=[1,5,6,7] del X[2]	X=[1,5,7]
Get **slice** of list (slicing)	<list>[start:stop:step]	X = [4,5,6,7] X[1:3]	[5,6]
Iterate over list (iteration)	for <var> in <list>:	X = [2,4,6] for a in X: print (a+1)	3 5 7

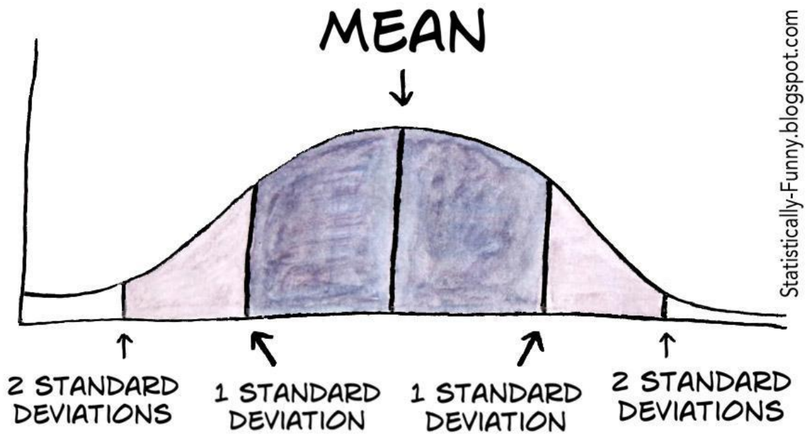
Common List Functions

Function	Syntax	Example	X
		X = [1,3,5,4,2]	
Add element to end	<list>.append(<item>)	X.append(6)	X=[1,3,5,4,2,6]
Sort list	<list>.sort()	X.sort()	X=[1,2,3,4,5,6]
Reverse items	<list>.reverse()	X.reverse()	X=[6,5,4,3,2,1]
Find position of item	<list>.index(<item>)	Y = X.index(4)	Y=2
Insert item into list	<list>.insert(<i>,<item>)	X.insert(2,2)	X=[6,5,2,4,3,2,1]
Count occurrences of item	<list>.count(<item>)	Y=X.count(2)	Y=2
Remove first occurrence of item	<list>.remove(<item>)	X.remove(2)	X=[6,5,4,3,2,1]
Remove and return specified item	<list>.pop(<i>)	Y=X.pop(1)	Y=5, X=[6,4,3,2,1]
Split string into list	<string>.split(<sep>)	“4,7,3”.split(“,”)	['4','7','3']
Join list into string	<sep>.join(<list>)	“-”.join(['4','7','3'])	“4-7-3”

Problem 1: Basic statistics program

- Write a program to output the median (x) and the standard deviation (s) of a series of values typed in by the user.

$$s = \sqrt{\frac{\sum(\bar{x} - x_i)^2}{n-1}}$$



A plot of a normal distribution (or bell-shaped curve) where each band has a width of 1 standard deviation

Problem 2

- Write a program to act as the “Magic 8-ball”.

The Magic 8-Ball is a toy used for fortune-telling or seeking advice, developed in the 1950s and manufactured by Mattel.

It is often used in fiction, often for humor related to its giving accurate, inaccurate, or otherwise statistically improbable answers. [Wikipedia]

Problem 3

Write a program to perform some common list functions without using the built-in functions:

- reverse
- index
  - Did this one already
- count

Problem 4

- Write a program to generate custom spam.
  - The user must enter a list of names and a message.
  - Then the program must print out a customised message, in each case with <name> replaced by the actual name.
  - (This is called “templating”.)
- e.g.

Congratulations <name> you have won R10 000 000! To claim your reward <name>, send your banking details to Ms Connie Art at another.mark@gmail.com. Don’t forget to include your bank login details!

Problem 5: The prefix-sum problem

- Given a list of integers, `input`, produce output where `output[i]` is the sum of `input[0]+input[1]+…+input[i]`

- Also called cumulative sum or scan.

- If the values are [1,2,5,3,4], the prefix sums are [1,3,8,11,15].

UNIVERSITY OF CAPE TOWN

IYUNIVESITHI YASEKAPA • UNIVERSITEIT VAN KAAPSTAD

SCHOOL OF IT

Introduction to Programming

UNIVERSITY OF CAPE TOWN
IYUNIVESITHI YASEKAPA • UNIVERSITEIT VAN KAAPSTAD

Arrays

2-D Arrays and Dictionaries

CSC1

Hussein Suleman
Department of Computer Science

2-D Arrays and Dictionaries

Hussein Suleman
Department of Computer Science
University of Cape Town

2-Dimensional Arrays

Each item in a list could itself be a list.

- For example:

```
X = [[1,2], [3,4]]
```

2-D arrays/lists are equivalent to matrices or grids.

- All operations work just as before, but every item is now a list.
- To access an item, remember that X[0] will give us a list, so X[0][0] gives us the item in position (0,0) of the grid.

2-Dimensional Arrays

X

[0][0]	[0][1]	[0][2]	[0][3]
[1][0]	[1][1]	[1][2]	[1][3]
[2][0]	[2][1]	[2][2]	[2][3]
[3][0]	[3][1]	[3][2]	[3][3]

Problem 6

- In computer graphics, a gradient fill is when the colour of an area gradually changes from one colour to another.
  - Write a program to output the result of a gradient fill on a 4x4 grid of pixels where the top-left pixel is 0 and the bottom-left is 6.

0	1	2	3
1	2	3	4
2	3	4	5
3	4	5	6

Dictionaries

Instead of a sequence with indices, Python also has a data structure with arbitrary index values and no order - this is called a dictionary.

- A dictionary is a set of key=value pairs.
- Dictionaries are very efficient for storing/retrieving values and mapping one list to another.
- Define as follows:

```
D={'a':'apples', 'b':'bananas', 'p':'pears'}
```

Common Dictionary Operations

Function	Syntax	Example	X
Checking for key	<key> in <dict>	X = {1:2} 1 in X	True
Access dictionary item	<dict>[<key>]	X = {1:2, 3:4} X[3]	4
Get keys	<dict>.keys()	X= {1:2, 3:4} X.keys()	[1,3]
Get values	<dict>.values()	X= {1:2, 3:4} X.values()	[2,4]
Delete item	del <dict>[<key>]	X= {1:2, 3:4} del X[3]	X={1:2}
Clear dictionary	<dict>.clear()	X = {1:2, 3:4} X.clear()	X={}
Iterate over keys	for <var> in <dict>:	X={1:'One',2:'Two',3:'Three'} for a in X: print (X[a])	One Two Three

Problem 7

- Count the number of times each unique word occurs in a sentence. E.g.

It was a bright cold day in April, and the clocks were striking thirteen. Winston Smith, his chin nuzzled into his breast in an effort to escape the vile wind, slipped quickly through the glass doors of Victory Mansions, though not quickly enough to prevent a swirl of gritty dust from entering along with him.

UNIVERSITY OF CAPE TOWN

IYUNIVESITHI YASEKAPA • UNIVERSITEIT VAN KAAPSTAD

SCHOOL OF IT

Introduction to Programming

UNIVERSITY OF CAPE TOWN
IYUNIVESITHI YASEKAPA • UNIVERSITEIT VAN KAAPSTAD

Arrays

Practical Data Structures

CSC1

Hussein Suleman
Department of Computer Science

Practical Data Structures

Hussein Suleman
Department of Computer Science
University of Cape Town

n-Dimensional Arrays

Lists can easily contain more than 2 dimensions.

- For example a 3-d structure can be:
  - X=[[[1,2],[3,4]], [[5,6],[7,8]], [[9,10],[11,12]]]
  - X[1][1][0]= ?

Suppose we want to store a list of students, with a list of courses for each student, and a list of test dates for each course, and a list of chapters per test, we could use the following 4-d list:

```
students = [['saleem',['csc1015f',['12 april',['5','6','7']]]]]
```

Generic structures

- Every data structure is generic. So just as it is possible to have n-dimensional lists, you can have dictionaries of lists, list of dictionaries, etc.

```
D = {'kayleigh':[12,23,31],
'callum':[45,54,55]}

E = [{'name':'palesa','year':1},
{'name':'luqmaan','year':1}]
```

Problem 8: Battleships

- Write a single-player Battleships game
  - The user is presented with a 10x10 grid with hidden warships and must guess the locations of these warships until all are hit.

Arrays in Functions

- Passing arrays to functions
  - In Python, the values of actual parameters cannot be changed in a function.
    - Arrays cannot be replaced.
  - HOWEVER, the contents of arrays can be changed.
    - This is a way of passing back multiple values from a function

Exercise: Scope and Arrays

What is the **exact output** of this code?

```
#scope.py - illustrating scope
def tester(myArr):
    print("In function, before append",myArr)
    myArr.append('a')
    print("In function, after append",myArr)
    myArr=['a','b','c']
    print("In function, after reassignment",myArr)

print()
arrMain=["one","two","three"]
print("Before function call", arrMain)
tester(arrMain)
print("After function call", arrMain)
```

Reminder: Scope and Lifetime

- Not all variables are accessible from all parts of our program, and not all variables exist for the same amount of time.
- Where a variable is accessible and how long it exists depend on how it is defined.
  - The part of a program where a variable is accessible or modifiable is called a **scope**.
  - The duration for which the variable exists is its **lifetime**.
- Formal parameters can also only be seen inside the function, even if they have the same name.

Reminder: Scope

- A variable that is defined in the main body of a file is called a **global variable**.
  - It will be visible throughout the file, and also inside any file that imports that file.
- A variable that is defined inside a function is local to that function.
  - It is accessible from the point at which it is defined until the end of the function and it exists for as long as the function is executing.
  - The parameter names in the function definition behave like local variables.
    - But they contain the values that we pass into the function when we call it.

Reminder: Global variables

- global / nonlocal
  - Python parameters can be declared as **global** in a function to indicate that the variable being used is actually declared outside any functions. This is not recommended in general.
  - **nonlocal** can be used similarly for nested functions.

Poll

What is scope?

- A – `where a variable has a value`
- B – `where a variable can be accessed/changed`
- C – `where a variable is stored`
- D – `where a variable overrides another`

Solution

What is scope?

- A – `where a variable has a value`
- B – `where a variable can be accessed/changed` 
- C – `where a variable is stored`
- D – `where a variable overrides another`

Poll

When can you read a global variable?

- A – `if the name is not overridden locally`
- B – `when you use the word nonlocal`
- C – `only when you use the prefix “global”`
- D – `when the variable is defined in a module`

Solution

When can you read a global variable?

- A – `if the name is not overridden locally` 
- B – `when you use the word nonlocal`
- C – `only when you use the prefix “global”`
- D – `when the variable is defined in a module`

Poll

When can you write to a global variable?

- A – `if the name is not overridden locally`
- B – `when you use the word nonlocal`
- C – `only when you use the prefix “global”`
- D – `when the variable is defined in a module`

Solution

When can you write to a global variable?

- A – `if the name is not overridden locally`
- B – `when you use the word nonlocal`
- C – `only when you use the prefix “global”` 
- D – `when the variable is defined in a module`

Exercise

Describe briefly, and in clear English, what the function Enigma returns.

```
def Enigma(lst,item):
    tmp=[]
    for i in lst:
        if i!= item:
            tmp.append(i)
    return tmp
```

Exercise

Write down the exact output:

```
def main():
    list1=[2,4,6,6,8,10]
    list2=["Skipper","Kowalski","Rico","Private","King Julian"]
    list3=[[1,2],[3,4],[5,6]]
    print(Enigma(list2,"King Julian"))
    print(Enigma(list1,9))


main()
```

UNIVERSITY OF CAPE TOWN

IYUNIVESITHI YASEKAPA • UNIVERSITEIT VAN KAAPSTAD

SCHOOL OF IT