

# GCSE

Practical  
programming  
skills in Python

## Two- dimensional lists



PG ONLINE

9

# Objectives

- Understand the nature of a 2D list
- Be able to use a 2D list to solve a problem

# Starter activity

- A hotel has five floors, including the ground floor
- What order would you choose if you had to clean all the rooms in this hotel?
  - Try to describe the algorithm using as few steps as possible

# Starter activity

1. Clean all the rooms on the ground floor
2. Then the first floor
3. Then the second floor
4. Then the third floor
5. Then the fourth floor



# Starter activity

- For each floor in the hotel:
  - For each room on that floor:
    - Clean that room

# One-dimensional lists

- A one-dimensional list allows you to store several values together in one place

names



0	1	2	3	4
"Alison"	"Bob"	"Carol"	"Dave"	"Edgar"

```
names[1] = "Bob"
```

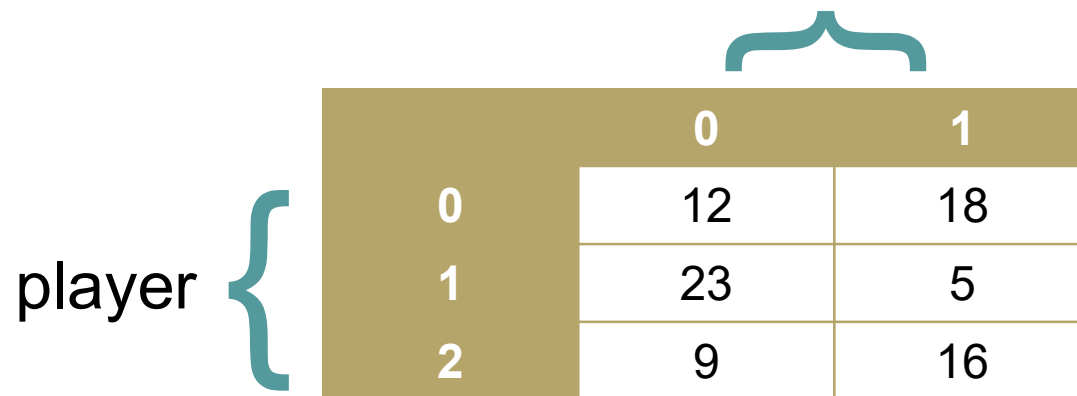
```
len(names) = 5
```

# Two-dimensional lists

- A two-dimensional list allows you to store data that would usually fit into a table

**highScores**

points in each game



	0	1
0	12	18
1	23	5
2	9	16

# Two-dimensional lists

- You address the row first (which player)
- Then the column (which go)

**highScores**                      points in each game (2<sup>nd</sup>)

		0	1
Player (1 <sup>st</sup> )	0	12	18
	1	23	5
	2	9	16



# Two-dimensional lists

- You address the row first (which player)
- Then the column (which go)

**highScores**                      points in each game (2<sup>nd</sup>)

		0	1
Player (1 <sup>st</sup> )	0	12	18
	1	23	5
	2	9	16

`highScores[1][0] =`

# Two-dimensional lists

- You address the row first (which player)
- Then the column (which go)

**highScores**                      points in each game (2<sup>nd</sup>)

		0	1
Player (1 <sup>st</sup> )	0	12	18
	1	23	5
	2	9	16

`highScores[1][0] =`

# Two-dimensional lists

- You address the row first (which player)
- Then the column (which go)

**highScores**                  points in each game (2<sup>nd</sup>)

		0	1
Player (1 <sup>st</sup> )	0	12	18
	1	23	5
	2	9	16

`highScores[1][0] =`

# Two-dimensional lists

- You address the row first (which player)
- Then the column (which go)

**highScores**                      points in each game (2<sup>nd</sup>)

		0	1
Player (1 <sup>st</sup> )	0	12	18
	1	23	5
	2	9	16

`highScores[1][0] = 23`

# Worksheet 9a

- Complete **Question 1**





# Creating a 2-D List

- Try the following code

```
highScores = [ [12,18], [23,5], [9,16] ]  
print(highScores)  
print(highScores[1][0])
```

	0	1
0	12	18
1	23	5
2	9	16

# Creating a 2-D List

- A 2-D list is basically a list of lists

```
highScores = [ [12,18], [23,5], [9,16] ]
```

                    └──┬──┘     └──┬──┘     └──┬──┘  
                    Player    Player    Player  
                    0            1            2

	0	1
0	12	18
1	23	5
2	9	16

# Appending to a 2-D List

- Append an extra list to add an extra row:

```
highScores = [ [12,18], [23,5], [9,16] ]  
highScores.append([20,20])
```

	0	1
0	12	18
1	23	5
2	9	16
3	20	20

# Worksheet 9a

- Complete **Questions 2 - 5**



# Finding the size of a list

- What will be the result of this code?

```
highScores = [ [12,18], [23,5], [9,16] ]  
print(len(highScores))
```

	0	1
0	12	18
1	23	5
2	9	16



# Finding the size of a list

- What will be the result of this code?

```
highScores = [ [12,18], [23,5], [9,16] ]  
print(len(highScores))
```

3 {

	0	1
0	12	18
1	23	5
2	9	16

# Finding the size of a list

- What will be the result of this code?

```
highScores = [ [12,18], [23,5], [9,16] ]  
print(len(highScores[0]))
```

	0	1
0	12	18
1	23	5
2	9	16

# Finding the size of a list

- What will be the result of this code?

```
highScores = [ [12,18], [23,5], [9,16] ]  
print(len(highScores[0]))
```

2

	0	1
0	12	18
1	23	5
2	9	16

# Worksheet 9b

- Complete **Question 1**



# Stepping through a list

- How can you print out each score for player 0?

```
highScores = [ [12,18] , [23,5] , [9,16] ]
```

	0	1
0	12	18
1	23	5
2	9	16



# Stepping through a list

- How can you print out each score for player 0?

```
highScores = [ [12,18] , [23,5] , [9,16] ]  
print(highScores[0][0])  
print(highScores[0][1])
```

	0	1
0	12	18
1	23	5
2	9	16

# Stepping through a list

- How can you print out each score for player 0?

```
highScores = [ [12,18] , [23,5] , [9,16] ]  
print(highScores[0][0])  
print(highScores[0][1])
```

- Can this be done more efficiently with a loop?

	0	1
0	12	18
1	23	5
2	9	16

# Stepping through a list

- How can you print out each score for player 0?

```
highScores = [ [12,18] , [23,5] , [9,16] ]  
for i in range(len(highScores[0])):  
    print(highScores[0][i])
```

	0	1
0	12	18
1	23	5
2	9	16

# Stepping through a list

- How can you print out each score for player 0?

```
highScores = [ [12,18] , [23,5] , [9,16] ]  
for i in range(len(highScores[0])):  
    print(highScores[0][i])
```

`len(highScores[0]) = 2`

i	target	value

	0	1
0	12	18
1	23	5
2	9	16

# Stepping through a list

- How can you print out each score for player 0?

```
highScores = [ [12,18] , [23,5] , [9,16] ]  
for i in range(len(highScores[0])):  
    print(highScores[0][i])
```

`len(highScores[0]) = 2`

i	target	value
0		

	0	1
0	12	18
1	23	5
2	9	16

# Stepping through a list

- How can you print out each score for player 0?

```
highScores = [ [12,18] , [23,5] , [9,16] ]  
for i in range(len(highScores[0])):  
    print(highScores[0][i])
```

`len(highScores[0]) = 2`

i	target	value
0	[0][0]	

	0	1
0	12	18
1	23	5
2	9	16

# Stepping through a list

- How can you print out each score for player 0?

```
highScores = [ [12,18] , [23,5] , [9,16] ]  
for i in range(len(highScores[0])):  
    print(highScores[0][i])
```

`len(highScores[0]) = 2`

i	target	value
0	[0][0]	12

	0	1
0	12	18
1	23	5
2	9	16



# Stepping through a list

- How can you print out each score for player 0?

```
highScores = [ [12,18] , [23,5] , [9,16] ]  
for i in range(len(highScores[0])):  
    print(highScores[0][i])
```

`len(highScores[0]) = 2`

i	target	value
0	[0][0]	12

	0	1
0	12	18
1	23	5
2	9	16

# Stepping through a list

- How can you print out each score for player 0?

```
highScores = [ [12,18] , [23,5] , [9,16] ]  
for i in range(len(highScores[0])):  
    print(highScores[0][i])
```

`len(highScores[0]) = 2`

i	target	value
0	[0][0]	12
1	[0][1]	

	0	1
0	12	18
1	23	5
2	9	16

# Stepping through a list

- How can you print out each score for player 0?

```
highScores = [ [12,18] , [23,5] , [9,16] ]  
for i in range(len(highScores[0])):  
    print(highScores[0][i])
```

`len(highScores[0]) = 2`

i	target	value
0	[0][0]	12
1	[0][1]	18

	0	1
0	12	18
1	23	5
2	9	16

# Worksheet 9b

- Complete **Questions 2 and 3**



# Stepping through a list

- How can you get the total score for player 0?

```
highScores = [ [12,18] , [23,5] , [9,16] ]  
for i in range(len(highScores[0])):  
    print(highScores[0][i])
```

	0	1
0	12	18
1	23	5
2	9	16

# Stepping through a list

- How can you get the total score for player 0?

```
total = 0
highScores = [ [12,18] , [23,5] , [9,16] ]
for i in range(len(highScores[0])):
    total = total + highScores[0][i]
print(total)
```

	0	1
0	12	18
1	23	5
2	9	16

# Worksheet 9b

- Complete **Questions 4 and 5**





# Sorting a list

- Suppose we have a 2-D list `gameScores`, with contents shown in the table below
- We want to sort the list in descending order of scores
  - How could you refer to Dave's name?
  - How could you refer to Phil's score?

	0	1
0	Dave	18
1	Christina	23
2	Phil	20

# Sorting a list

- To create this 2D list, use the following statement:

```
gameScores =  
[["Dave",18],["Christina",23],["Phil",20]]
```

	0	1
0	Dave	18
1	Christina	23
2	Phil	20

# A bubble sort

Repeat this 'numRows' - 1 times:

	0	1
0	Dave	18
1	Christina	23
2	Phil	20

# A bubble sort

Repeat this 'numRows' - 1 times:

Check if scores in the next two rows  
are in the wrong order

	0	1
0	Dave	18
1	Christina	23
2	Phil	20

# A bubble sort

Repeat this 'numRows' - 1 times:

If the two scores are in the wrong order

Swap the rows

	0	1
0	Dave	18
1	Christina	23
2	Phil	20

# A bubble sort

Repeat the whole process 'numRows - 1' times:

Repeat this 'numRows' - 1 times:

If two scores are in the wrong order

Swap the rows

	0	1
0	Dave	18
1	Christina	23
2	Phil	20

# A bubble sort algorithm

```
numRows = len(gameScores)
for i in range(numRows-1):
    for j in range(numRows-1):
        #compare the scores in next 2 rows
        if gameScores[j][1] < gameScores [j+1][1]:
            # swap the rows
            temp = gameScores[j]
            gameScores[j]= gameScores[j+1]
            gameScores[j+1] = temp
```



# Sorting a 2D list

- You can use a lambda function to sort a 2D list
  - A lambda function allows you to specify which column to sort on
  - data is a temporary variable to store each row
  - data[1] means that the sort is done on column 1

```
highScores =  
sorted(highScores, key=lambda data:data[1])
```

	0	1
0	Dave	18
1	Phil	20
2	Christina	23

# Sorting a 2D list in descending order

- To sort in descending order of score:

```
highScores = sorted(highScores, key=lambda  
data:data[1], reverse = True)
```

- How would you sort on name in reverse alphabetical order?

	0	1
0	Dave	18
1	Phil	20
2	Christina	23

# Worksheet 9c

- Complete **Worksheet 9c**



# Plenary

- Which statement below would print the value highlighted in black?
  - `print(values[0][1])`
  - `print(values[1][0])`
  - `print(values[0,1])`
  - `print(values[1,0])`

	0	1
0	12	18
1	<b>23</b>	20
2	9	23

# Plenary

- What code would print the value highlighted in black?
  - `print(values[0][1])`
  - `print(values[1][0])`
  - `print(values[0,1])`
  - `print(values[1,0])`

	0	1
0	12	18
1	23	20
2	9	23

## Copyright

© 2017 PG Online Limited

The contents of this unit are protected by copyright.

This unit and all the worksheets, PowerPoint presentations, teaching guides and other associated files distributed with it are supplied to you by PG Online Limited under licence and may be used and copied by you only in accordance with the terms of the licence. Except as expressly permitted by the licence, no part of the materials distributed with this unit may be used, reproduced, stored in a retrieval system, or transmitted, in any form or by any means, electronic or otherwise, without the prior written permission of PG Online Limited.

## Licence agreement

This is a legal agreement between you, the end user, and PG Online Limited. This unit and all the worksheets, PowerPoint presentations, teaching guides and other associated files distributed with it is licensed, not sold, to you by PG Online Limited for use under the terms of the licence.

The materials distributed with this unit may be freely copied and used by members of a single institution on a single site only. You are not permitted to share in any way any of the materials or part of the materials with any third party, including users on another site or individuals who are members of a separate institution. You acknowledge that the materials must remain with you, the licencing institution, and no part of the materials may be transferred to another institution. You also agree not to procure, authorise, encourage, facilitate or enable any third party to reproduce these materials in whole or in part without the prior permission of PG Online Limited.