**CS 161 Python Week 8 in Class Exercise Instructions**

1. Use the following lists to make predictions of the instructions then check for the actual. Do your prediction first then use Spyder to check for the actual. If there’s no output then write what the assignment would be, whether it’s True or False or what it evaluates to. If there’s an error then write “error”.

spreadsheet\_list = [[‘Name’, ‘Age’, ‘GPA’], [‘Bill’, 25, 3.55], [‘Rick’, 26, 4.00]]

my\_list = [1, ‘a’, 3.14159, True]

a\_list = [1, 2, 3]

b\_list = [‘a’, ‘b’, ‘c’]

int\_list = [1, 2, 3, 4, 5]

float\_list = [1.0, 2.0, 3.0, 4.0, 5.0]

str\_list = [‘a’, ‘b’, ‘c’, ‘d’, ‘e’]

nested\_list = [int\_list, float\_list, str\_list]

new\_list = [1, 2, ‘a’, ‘z’]

c\_list = [1, 12, 5, 8]

d\_list = [20, 2]

***NOTE: To avoid unexpected errors, re-type in Spyder rather than copying and pasting***

|  |  |  |
| --- | --- | --- |
| **Instruction** | **Prediction** | **Actual** |
| print(spreadsheet\_list[1]) | [‘Bill’,25,3.55] | [‘Bill’,25,3.55] |
| row = spreadsheet\_list[1] | Creates row and assigns it the second list – same as first instruction | Assigned new variable row the list from first instruction |
| print(row) | Prints [‘Bill’,25,3.55] | Prints [‘Bill’,25,3.55] |
| column = row[2] | Assigns column 3.55 | Assigns column 3.55 |
| print(column) | Prints 3.55 | Printed 3.55 |
| print(spreadsheet\_list[1][2]) | Prints 3.55 | Printed 3.55 |
| print(my\_list[1]) | Prints a | Printed ‘a’ |
| print(my\_list[-1]) | Prints True | Printed True |
| my\_list2 = my\_list[:] | Assigns my\_list2 all of my\_list | Assigned all of my\_list into my\_list2 |
| print(my\_list2) | Prints [1,’a’,3.14159,True] | Printed [1,’a’,3.14159,True] |
| print(my\_list[: 3 : 2]) | Prints [‘a’] | Printed[1, 3.14159] |
| print(my\_list[: : 2]) | Prints[1,3.14159] | Printed[1, 3.14159] |
| print(my\_list[2:]) | Prints[‘a’,3.14159,True] | Printed[3.14159,True] |
| my\_list[10] | Index error will happen because the list isn’t that long | List index out of range |
| concat\_list = a\_list + b\_list | Concat\_list is assigned: [1,2,3,’a’,’b’,’c’] | Assigned new variable concat\_list: [1,2,3,’a’,’b’,’c’] |
| print(concat\_list) | Prints [1,2,3,’a’,’b’,’c’] | Printed [1,2,3,’a’,’b’,’c’] |
| concat\_list = b\_list + a\_list | Reassigns concat\_list: [’a’,’b’,’c’,1,2,3] | Reassigned concat\_list: [’a’,’b’,’c’,1,2,3] |
| print(concat\_list) | Prints [’a’,’b’,’c’,1,2,3] | Printed [’a’,’b’,’c’,1,2,3] |
| rep\_list = a\_list \* 3 | Assigns new list rep\_list: [1,2,3,1,2,3,1,2,3] | Assigned rep\_list: [1,2,3,1,2,3,1,2,3] |
| print([1, 2, 3, 4] < [1, 2, 3, 0]) | False | False |
| print([1, 2, 3, 4] < [1, 2, 3, 4, 0]) | True | True |
| 1 in a\_list | True | True |
| 1 in b\_list | False | False |
| [1,2,’one’,’two’] < [3, 4, 5, 6] | True | True |
| [1,2,’one’,’two’] < [1, 2, 5, 6] | False | Doesn’t like this one, says ‘<’ not supported between instances of ‘srt’ and ‘int’ |
| len(int\_list) | 5 | 5 |
| len(nested\_list) | 3 | 3 |
| min(float\_list) | 1.0 | 1.0 |
| min(str\_list) | ‘a’ | ‘a’ |
| max(str\_list) | ‘e’ | ‘e’ |
| sum(int\_list) | 15 | 15 |
| sum(str\_list) | invalid | Unsupported operand type(s) |
| min(nested\_list) | invalid | ‘<’ not supported between instances of ‘str’ and ‘int’ |
| for item in my\_list:  print(item ,end = ' ') | Prints 1 a 3.14159 True | Printed 1 a 3.14159 True |
| new\_list[0] = True | Reassigns index 0 to True | Reassigned index 0 to True |
| print(new\_list) | Prints [True, 2, ‘a’, ‘z’] | Printed [True, 2, ‘a’, ‘z’] |
| new\_list[-1] = 7 | Reassigns new\_list[-1] (last variable) to 7 | Reassigned last variable to 7 |
| print(new\_list) | Prints [True, 2, ‘a’, 7] | Printed [True, 2, ‘a’, 7] |
| new\_list[:2] = [27] | Error invalid | Reassigned new\_list[27, ‘a’, 7] |
| print(new\_list) | Prints [27, ‘a’, 7] | Printed [27, ‘a’, 7] |
| new\_list[:] = [5, 6, 7, 8] | New\_list reassigned [5,6,7,8] | Reassigned new\_list [5,6,7,8] |
| print(new\_list) | Prints [5,6,7,8] | Printed [5,6,7,8] |
| new\_list[2:] = 'abc' | Reassigns new\_list: [5, ’abc’] | Reassigned new\_list: [5,6,’a’,’b’,’c’] |
| print(new\_list) | Prints [5,6,’a’,’b’,’c’] | Printed [5,6,’a’,’b’,’c’] |
| new\_list[:2] = 15 | Reassigns new\_list: [15,’a’,’b’,’c’] | Error: can only assign an iterable |
| print(new\_list) | Prints [5,6,’a’,’b’,’c’] | Printed [5,6,’a’,’b’,’c’] |
| c\_list.append(17) | Adds 17 onto the end of c\_list | Added 17 to the end of c\_list |
| print(c\_list) | Prints [1,12,5,8,17] | Printed [1,12,5,8,17] |
| c\_list.append([40, 50, 60]) | Adds [40,50,60] to the end of c\_list: [1,12,5,8,17, [40,50,60]] | Changed c\_list to: [1,12,5,8,17, [40,50,60]] |
| print(c\_list) | Prints [1,12,5,8,17, [40,50,60]] | Printed [1,12,5,8,17, [40,50,60]] |
| print(d\_list) | Prints [20,2] | Printed [20,2] |
| c\_list.extend(d\_list) | Adds d\_list onto the end of c\_list | Added d\_list to the end of c\_list |
| print(c\_list) | Prints [1, 12, 5, 8, 17, [40, 50, 60], 20, 2] | Printed [1, 12, 5, 8, 17, [40, 50, 60], 20, 2] |
| c\_list.insert(3,30) | Puts 30 at index 3 of c\_list | Put 30 at index 3 of c\_list |
| print(c\_list) | Prints [1, 12, 5, 30, 8, 17, [40, 50, 60], 20, 2] | Printed [1, 12, 5, 30, 8, 17, [40, 50, 60], 20, 2] |
| c\_list.remove(8) | Removes 8 from c\_list | Removed 8 from c\_list |
| print(c\_list) | Prints [1, 12, 5, 30, 17, [40, 50, 60], 20, 2] | Printed [1, 12, 5, 30, 17, [40, 50, 60], 20, 2] |
| c\_list.pop() | Removes the last value from c\_list | Removed 2 from c\_list |
| print(c\_list) | Prints [1, 12, 5, 30, 17, [40, 50, 60], 20] | Printed [1, 12, 5, 30, 17, [40, 50, 60], 20] |
| c\_list.index(17) | Prints 4 | Printed 4 |
| c\_list.count(5) | Prints 1 | Printed 1 |
| c\_list.remove([40, 50, 60]) | Removes the list [40,50,60] from c\_list | Removed [40, 50, 60] from c\_list |
| print(c\_list) | Prints [1, 12, 5, 30, 17, 20] | Printed [1, 12, 5, 30, 17, 20] |
| c\_list.sort() | Organizes the list from least to greatest | Organized the list from least to greatest |
| print(c\_list) | Prints [1, 5, 12, 17, 20, 30] | Printed [1, 5, 12, 17, 20, 30] |
| c\_list.reverse() | Flips the list around to be greatest to least | Flipped the list around to be greatest to least |
| print(c\_list) | Prints [30, 20, 17, 12, 5, 1] | Printed [30, 20, 17, 12, 5, 1] |
| result = 'this is a test'.split() | Assigns result:[‘this’,’is’,’a’,’test’] | Assigned new variable result:[‘this’,’is’,’a’,’test’] |
| print(result) | Prints [‘this’,’is’,’a’,’test’] | Printed [‘this’,’is’,’a’,’test’] |
| result1 = 'f1, f2, f3, f4'.split(',') | Assigns new variable result1:[‘f1’,’ f2’,’ f3’,’ f4’] | Assigned new variable result1:[‘f1’,’ f2’,’ f3’,’ f4’] |
| print(result1) | Prints [‘f1’,’ f2’,’ f3’,’ f4’] | Printed [‘f1’,’ f2’,’ f3’,’ f4’] |
| item1, item2, item3 = [11, 12,1 3] | Assigns item1=11, item2=12, item3=’1 3’ | Invalid syntax |
| print(item2) | Error no variable | Name ‘item2’ is not defined |
| f1, f2, f3 = 'Python is great'.split() | Assigns f1:Python, f2:is, f3:great | Assigned f1:Python, f2:is, f3:great |
| print(f3) | Prints great | Printed ‘great’ |
| item1, item2 = [1, 2, 3] | Error too many values | Too many values to unpack |
| item1, item2, item3 = [1, 2] | Error too many items | Not enough values to unpack |
| my\_list2 = [27,56,4,18] | Assigns my\_list2 [27,56,4,18] | Assigned new variable my\_list2 [27,56,4,18] |
| print(sorted(my\_list2)) | Sorts my\_list2 and prints it in lest to greatest order | Sorted my\_list2 and prints it in lest to greatest order |
| print(my\_list2) | Prints [27,56,4,18] | Printed [27,56,4,18] |
| print(my\_list2.sort()) | Not a clue | Printed None |
| What is the difference between sort() and sorted()? | Sorted organizes the item from least to greatest and sort organizes based on a parameter |  |
| Write an instruction using spreadsheet\_list to print the following:  [‘Name’, ‘Age’, ‘GPA’] | Print(spreadsheet\_list[0]) |  |
| Write an instruction using spreadsheet\_list to print Rick | Print(spreadsheet\_list[2][0]) |  |

1. Complete zyBooks 8.18 – 8.24