

18/6/25 Python Training Assignment

SECTION A - TUPLES

1. CREATE TUPLE

```
my_tuple = ("Harcini", 20, "AI", "DS")
print("Created Tuple:", my_tuple)
```

2. ACCESS TUPLE

```
print("First item:", my_tuple[0])
print("Last item:", my_tuple[-1])
```

3. LOOP TUPLE

```
print("Looping through tuple:")
for item in my_tuple:
```

```
    print(item)
```

4. RANGE TUPLE

```
for i in range(len(my_tuple)):
    print(f"Item at index {i}:", my_tuple[i])
```

5. SLICE TUPLE

```
print("Slice 1 to 3:", my_tuple[1:3])
print("Slice from beginning:",
      my_tuple[:2])
print("Slice to end:", my_tuple[2:])
```

6. UPDATE ITEMS IN TUPLE

```
temp = list(my_tuple)
temp[1] = 21
my_tuple = tuple(temp)
print("Updated Tuple:", my_tuple)
```

```
PS C:\Users\harcini\OneDrive\Desktop\coding python training> & C:/Us
coding python training/tup.py"
Created Tuple: ('Harcini', 20, 'AI', 'DS')
First item: Harcini
Last item: DS
Looping through tuple:
Harcini
20
AI
DS
Item at index 0: Harcini
Item at index 1: 20
Item at index 2: AI
Item at index 3: DS
Slice 1 to 3: (20, 'AI')
Slice from beginning: ('Harcini', 20)
Slice to end: ('AI', 'DS')
Updated Tuple: ('Harcini', 21, 'AI', 'DS')
PS C:\Users\harcini\OneDrive\Desktop\coding python training>
```

SECTION B – LIST

Part -A

1. DECLARE LIST

```
fruits = ["apple", "banana", "cherry"]  
print("Original List:", fruits)
```

2. SORT LIST

```
fruits.sort()  
print("Sorted List:", fruits)
```

3. INSERT LIST

```
fruits.insert(1, "orange")  
print("After Insert:", fruits)
```

4. REMOVE LIST

```
fruits.remove("banana")  
print("After Remove:", fruits)
```

5. JOIN LIST

```
more_fruits = ["grape", "mango"]  
combined = fruits + more_fruits  
print("After Joining:", combined)
```

6. CHANGE LIST

```
combined[0] = "kiwi"  
print("After Changing:", combined)
```

7. Access list

```
print("First item:", combined[0])  
print("Last item:", combined[-1])
```

8. LOOP LIST

```
print("Looping:")  
for fruit in combined:
```

```
    print(fruit)
```

9. COPY LIST

```
copy_list = combined.copy()  
print("Copied List:", copy_list)
```

```
coding python training/H.PY"  
Original List: ['apple', 'banana', 'cherry']  
Sorted List: ['apple', 'banana', 'cherry']  
After Insert: ['apple', 'orange', 'banana', 'cherry']  
After Remove: ['apple', 'orange', 'cherry']  
After Joining: ['apple', 'orange', 'cherry', 'grape', 'mango']  
After Changing: ['kiwi', 'orange', 'cherry', 'grape', 'mango']  
First item: kiwi  
Last item: mango  
Looping:  
kiwi  
orange  
cherry  
grape  
mango  
Copied List: ['kiwi', 'orange', 'cherry', 'grape', 'mango']  
PS C:\Users\harci\OneDrive\Desktop\coding python training> |
```

10. STRING METHODS IN LIST

Original list of names

```
names = ["harcini", "Vijaya", "heLLo", "WORLD"]  
print("Original List:", names)
```

Convert all to uppercase

```
upper_names = [name.upper() for name in names]  
print("Uppercase:", upper_names)
```

Convert all to lowercase

```
lower_names = [name.lower() for name in names]  
print("Lowercase:", lower_names)
```

Capitalize first letter only (rest lowercase)

```
capitalized_names = [name.capitalize() for name in names]  
print("Capitalized:", capitalized_names)
```

Title case (first letter of each word in uppercase)

```
title_names = [name.title() for name in names]  
print("Title Case:", title_names)
```

Swap case (uppercase becomes lowercase and vice versa)

```
swapped_names = [name.swapcase() for name in names]  
print(" Swapcase:", swapped_names)
```

```
coding python training/S.PY"
```

```
Original List: ['harcini', 'Vijaya', 'heLLo', 'WORLD']
```

```
Uppercase: ['HARCINI', 'VIJAYA', 'HELLO', 'WORLD']
```

```
Lowercase: ['harcini', 'vijaya', 'hello', 'world']
```

```
Capitalized: ['Harcini', 'Vijaya', 'Hello', 'World']
```

```
Title Case: ['Harcini', 'Vijaya', 'Hello', 'World']
```

```
Swapcase: ['HARCINI', 'vIJAYA', 'HEllo', 'world']
```

```
PS C:\Users\harci\OneDrive\Desktop\coding python training> █
```

Part-B

1. CREATES A LIST

```
fruits = ["apple", "banana", "cherry", "mango"]  
print("Original list:", fruits)
```

2. PRINTS A SPECIFIC INDEX

```
print("Item at index 2:", fruits[2])
```

3. CHANGES AN ITEM

```
fruits[1] = "orange"  
print("After changing index 1:", fruits)
```

4. APPENDS IN MULTIPLE WAYS

a. Using append()

```
fruits.append("grape")  
print("After append():", fruits)
```

b. Using insert(index, value)

```
fruits.insert(2, "kiwi") # Inserts 'kiwi' at index 2  
print("After insert():", fruits)
```

c. Using + operator

```
more_fruits = ["pineapple", "papaya"]  
fruits = fruits + more_fruits  
print("After + operator:", fruits)
```

d. Using extend()

```
fruits.extend(["melon", "pear"])  
print("After extend():", fruits)
```

5. Removes items using del, remove(), clear()

5. Remove items in all 3 ways

a. Using del to delete by index

```
del fruits[0]  
print("After del:", fruits)
```

b. Using remove() to remove by value

```
fruits.remove("mango")
```

```
print("After remove():", fruits)
```

c. Using clear() to remove all items

```
fruits.clear()
```

```
print("After clear():", fruits)
```

```
coding python training/list.py"
Original list: ['apple', 'banana', 'cherry', 'mango']
Item at index 2: cherry
After changing index 1: ['apple', 'orange', 'cherry', 'mango']
After append(): ['apple', 'orange', 'cherry', 'mango', 'grape']
After insert(): ['apple', 'orange', 'kiwi', 'cherry', 'mango', 'grape']
After + operator: ['apple', 'orange', 'kiwi', 'cherry', 'mango', 'grape', 'pineapple', 'papaya']
After extend(): ['apple', 'orange', 'kiwi', 'cherry', 'mango', 'grape', 'pineapple', 'papaya', 'melon', 'pear']
After del: ['orange', 'kiwi', 'cherry', 'mango', 'grape', 'pineapple', 'papaya', 'melon', 'pear']
After remove(): ['orange', 'kiwi', 'cherry', 'grape', 'pineapple', 'papaya', 'melon', 'pear']
After clear(): []
PS C:\Users\harci\OneDrive\Desktop\coding python training>
```

SECTION-C STRING

1. ESCAPE CHARACTERS

Escape characters demo using "Harcini"

```
print('It\'s Harcini\'s favorite book.')
```

```
print("Harcini said, \"I love Python!\")
```

```
print("The path is C:\\Users\\Harcini\\Desktop
```

Newline (\n)

```
print("Welcome Harcini!\nYou have logged in
successfully.")
```

**# Carriage return (\r) - replaces start of line
with what's after \r**

```
print("Harcini is awesome!\rWow!")
```

```
coding python training/es.py"
It's Harcini's favorite book.
Harcini said, "I love Python!"
The path is C:\Users\Harcini\Desktop
Welcome Harcini!
You have logged in successfully.
Wow!ini is awesome!
Name: Harcini Age: 21
Harcinix
Hello
Harcini
Alert! Harcini
Harcini
Harcini
Harcinix
Harcini
PS C:\Users\harci\OneDrive\Desktop\coding py
```

Tab (\t)

```
print("Name:\tHarcini\tAge:\t21")
```

Backspace (\b) - deletes the last character

```
print("Harcinix\b")
```

Octal representation of "Harcini"

```
print("\110\141\162\143\151\156\151")
```

Hexadecimal representation of "Harcini"

```
print("\x48\x61\x72\x63\x69\x6e\x69")
```

SECTION-D CONTROL STRUCTURE

1. TASK:- KID, TEEN, ADULT, OLD KID <12 TEEN :< KID TO 18 ADULT 18 TO 60

```
if age < 12:
    print("You are a Kid ")
elif age < 18:
    print("You are a Teen ")
elif age <= 60:
    print("You are an Adult ")
print("You are Old ")
```

```
PS C:\Users\harci\OneDrive\Desktop\coding\python training\control.py"
Enter your age: 21
You are an Adult
PS C:\Users\harci\OneDrive\Desktop\coding\python training\control.py"
```

2.STAR PYRAMID

```
rows = 5
for i in range(rows):
    spaces = ' ' * (rows - i - 1)
    stars = '*' * (2 * i + 1)
    print(spaces + stars)
```

```
PS C:\Users\harci\OneDrive\Desktop\coding\python training\pyramid.py"
*
***
*****
*****
*****
*****
PS C:\Users\harci\OneDrive\Desktop\coding\python training\pyramid.py"
```

3.ALPHABET PYRAMID

```
rows = 5
for i in range(rows):
    chars = ''.join(chr(65 + j) for j in range(i + 1))
    print(chars.center(2 * rows - 1))
```

```

  A
 A B
A B C
A B C D
A B C D E
PS C:\Users\harci\OneDrive\De
coding python training/pyram
```

4.NUMBER PYRAMID

```
rows = 5
for i in range(rows):
    numbers = "".join(str(j) for j in range(1, 2 * i + 2, 1))
    print(numbers.center(2 * rows - 1))
```

```

  1
 123
12345
1234567
123456789
PS C:\Users\harci\One
```

5.INVERTED STAR PYRAMID

```
rows = 5
for i in range(rows):
    spaces = ' ' * i
    stars = '*' * (2 * (rows - i) - 1)
    print(spaces + stars)
```

```

*****
 *****
  *****
   *****
    *****
     *
PS C:\Users\harci\OneDrive\D
```

6.BUTTERFLY PATTERN

```
rows = 5
for i in range(1, rows + 1):
    stars = '*' * i
    spaces = ' ' * (2 * (rows - i))
    print(stars + spaces + stars)
for i in range(rows, 0, -1):
    stars = '*' * i
    spaces = ' ' * (2 * (rows - i))
    print(stars + spaces + stars)
```

```

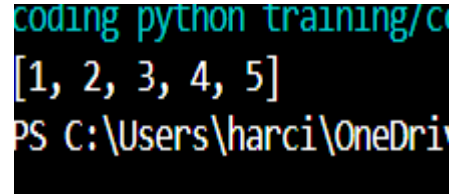
*           *
**          **
***         ***
****        ****
*****       *****
*****       *****
****        ****
***         ***
**          **
*           *
PS C:\Users\harci\OneDrive\Des
```

Python Coding Challenge Topic: List, Tuple, Dictionary, Set

Q1. Write a Python program to remove all duplicates from a list without using the set() function.

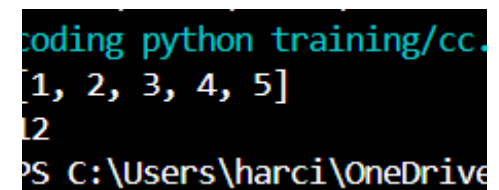
Input Example: [1, 2, 2, 3, 4, 4, 5] Output: [1, 2, 3, 4, 5]

```
lst = [1, 2, 2, 3, 4, 4, 5]
unique = []
for item in lst:
    if item not in unique:
        unique.append(item)
print(unique)
```



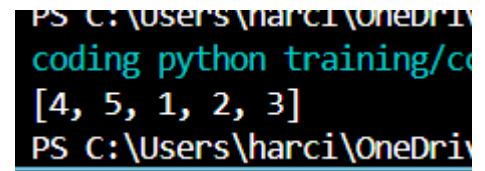
Q2. Given a list of integers, write a program to find the second highest unique number. Input Example: [12, 5, 9, 21, 21, 3] Output: 12

```
lst = [12, 5, 9, 21, 21, 3]
unique = list(set(lst))
unique.sort(reverse=True)
print(unique[1])
```



Q3. Rotate a list to the right by k positions. Input: List = [1, 2, 3, 4, 5], k = 2 Output: [4, 5, 1, 2, 3]

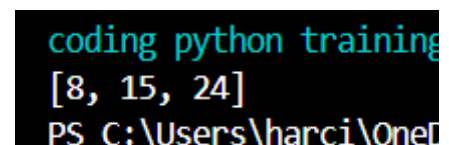
```
lst = [1, 2, 3, 4, 5]
k = 2
k = k % len(lst) # In case k > len
rotated = lst[-k:] + lst[:-k]
print(rotated)
```



Q4. Write a Python program to multiply the elements of each tuple in a list of tuples and return a new list

Input: [(2, 4), (3, 5), (4, 6)] Output: [8, 15, 24]

```
tpl_list = [(2, 4), (3, 5), (4, 6)]
result = [a * b for a, b in tpl_list]
print(result)
```



Q5. Given a tuple of integers, write a program to count how many times each element occurs. Input: (1, 2, 2, 3, 1, 4, 2) Output: {1: 2, 2: 3, 3: 1, 4: 1}

```
tpl = (1, 2, 2, 3, 1, 4, 2)
freq = {}
for item in tpl:
    freq[item] = freq.get(item, 0) + 1
print(freq)
```

```
PS C:\Users\harci\OneDrive\
coding python training/cc.py
{1: 2, 2: 3, 3: 1, 4: 1}
PS C:\Users\harci\OneDrive\
```

Q6. Write a Python program to count the frequency of each character in a string using a dictionary. Input: 'banana' Output: {'b': 1, 'a': 3, 'n': 2}

```
text = 'banana'
freq = {}
for char in text:
    freq[char] = freq.get(char, 0) + 1
print(freq)
```

```
coding python training/cc.py
{'b': 1, 'a': 3, 'n': 2}
PS C:\Users\harci\OneDrive\
```

Q7. Merge two dictionaries such that common keys have their values summed.

Input: {'apple': 10, 'banana': 5}, {'banana': 3, 'orange': 7} Output: {'apple': 10, 'banana': 8, 'orange': 7}

```
d1 = {'apple': 10, 'banana': 5}
d2 = {'banana': 3, 'orange': 7}
merged = d1.copy()
for key, value in d2.items():
    merged[key] = merged.get(key, 0) + value
print(merged)
```

```
coding python training/cc.py"
{'apple': 10, 'banana': 8, 'orange': 7}
PS C:\Users\harci\OneDrive\Desktop\codin
```

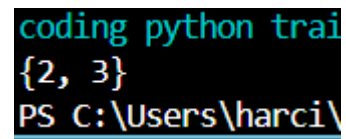
Q8. Given a dictionary of student names and their marks, print the name(s) of the student(s) with the highest marks. Input: {'Alice': 85, 'Bob': 92, 'Carol': 92} Output: ['Bob', 'Carol']

```
marks = {'Alice': 85, 'Bob': 92, 'Carol': 92}
max_score = max(marks.values())
top_students = [name for name, score in marks.items() if score ==
max_score]
print(top_students)
```

```
PS C:\Users\harci\OneDrive\
coding python training/cc
['Bob', 'Carol']
PS C:\Users\harci\OneDrive\
```

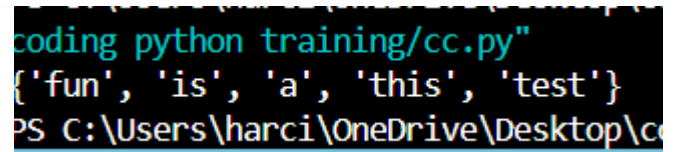
Q9. Write a Python program to find all common elements among three lists using set operations. Input: [1, 2, 3], [2, 3, 4], [3, 2, 5] Output: {2, 3}

```
a = [1, 2, 3]
b = [2, 3, 4]
c = [3, 2, 5]
common = set(a) & set(b) & set(c)
print(common)
```

A terminal window showing the execution of a Python program. The prompt is 'coding python trai'. The output is '{2, 3}'. The prompt is 'PS C:\Users\harci\'.

Q10. From a sentence entered by the user, extract and display all unique words using a set. Input: 'this is a test this is fun' Output: {'this', 'is', 'a', 'test', 'fun'}

```
sentence = 'this is a test this is fun'
words = set(sentence.split())
print(words)
```

A terminal window showing the execution of a Python program. The prompt is 'coding python training/cc.py'. The output is {'fun', 'is', 'a', 'this', 'test'}. The prompt is 'PS C:\Users\harci\OneDrive\Desktop\co'.