```
# ------------------------------------
# Class 8: Hands-on exercise using Data Structures
# ------------------------------------

customer1 = ["Taha", "03000000000", 67, 25]
customer2 = ["Nasir", "03010000000", 55, 20, "xyz@gmail.com"]

#acces data
customer1[4]
Traceback (most recent call last):
  File "<pyshell#9>", line 1, in <module>
    customer1[4]
IndexError: list index out of range

# Dictionary
customer1 = {
    "Name": "Taha",
    "Phone": "03000000000",
    "weight": 67,
    "Age": 25
    }
customer2 = {
    "Name": "Nasir",
    "Phone": "03010000000",
    "weight": 55,
    "Age": 20,
    "Email": "xyz@gmail.com"
    }

customer1.get("Age", "Record not found")
25
customer1.get("Email", "Record not found")
'Record not found'
```

```
customer1.items()
dict_items([('Name', 'Taha'), ('Phone', '03000000000'), ('weight', 67), ('Age
', 25)])

for key, value in customer1.items():
    print(f"{key} --> {value}")


Name --> Taha
Phone --> 03000000000
weight --> 67
Age --> 25

# fromkeys()
keys = ["a", "b", "c"]
my_dict = dict.fromkeys(keys, 0)
my_dict
{'a': 0, 'b': 0, 'c': 0}
```

```
# Task:
# 1. Create a dictionary and Add student records to it (NAme, Phone, Email)
# 2. Update Email and phone npo
# 3. add Student course into dictionary
# 4. Delete Phone from dictiojary
# 5. get (key, value) pairs from dictionary (.items())

# 1
student = {
    "Name": "Ahsan",
    "Phone": "03000000000",
    "Email": "xyz@gmail.com"
    }

# 2
student.update({"Email": "XYZ@gmail.com", "Phone": "03020000000"})

student
{'Name': 'Ahsan', 'Phone': '03020000000', 'Email': 'XYZ@gmail.com'}

# 3
student["course"] = "AI and DS"

student
{'Name': 'Ahsan', 'Phone': '03020000000', 'Email': 'XYZ@gmail.com', 'course':
'AI and DS'}

# 4
del student['Phone']

student
{'Name': 'Ahsan', 'Email': 'XYZ@gmail.com', 'course': 'AI and DS'}
```

```
for key, value in student.items():
    print(f"{key} --> {value}")


Name --> Ahsan
Email --> XYZ@gmail.com
course --> AI and DS
```

- **List comprehension:**

List comprehension is a concise way to create a new list in Python by writing the logic in a single line with loops and conditions.

**Syntax:**

*[expression **for** item **in** iterable **if** condition]*

- **Lambda function:**

A lambda function in Python is a short function without name, used to do simple task in one line.

Syntax:

*lambda arguments : expression*

Note: The expression is automatically returned when the lambda is executed.

Lambda function makes code concise and readable for simple operation , but for bigger task always used *def.*

```
# lambda function
# Example 1: lambda vs normal function
def add(x,y):
    return x+y

# lambda function
add_lambda = lambda x,y: x+y

print(add(3,4))
7
print(add_lambda(3,4))
7

# Example 2: using with sort()
students = [["Aqeel", "qw321", 78], ["Sara", "we432", 99]]
students.sort(key = lambda x:x[2])
print(students)
[['Aqeel', 'qw321', 78], ['Sara', 'we432', 99]]
students.sort(key = lambda x:x[2], reverse=True)
print(students)
[['Sara', 'we432', 99], ['Aqeel', 'qw321', 78]]
```