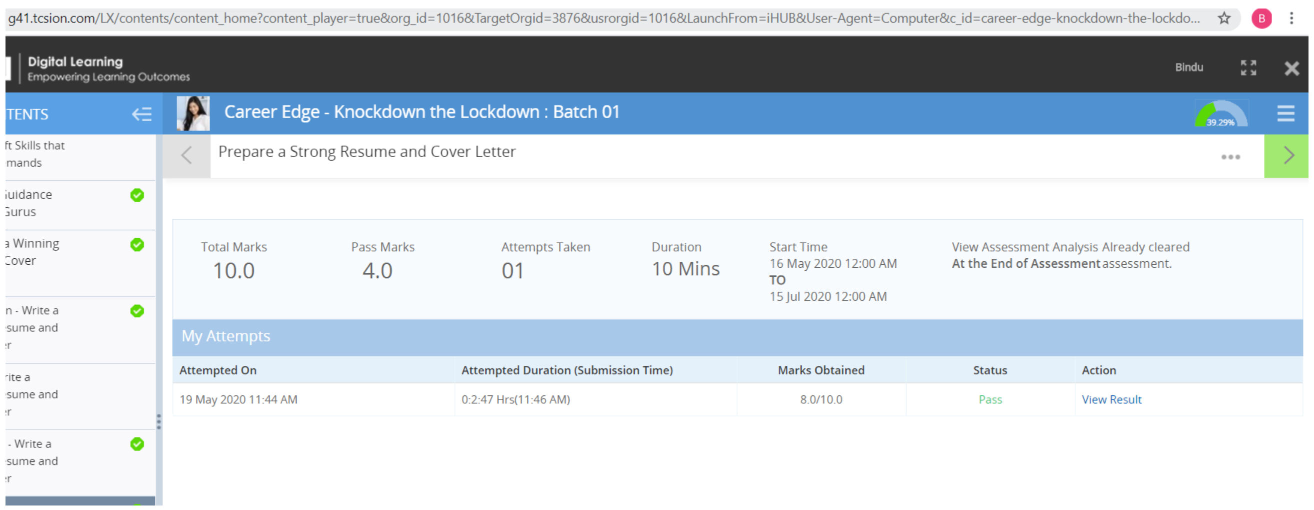
DAILY ASSESSMENT FORMAT

Date:	**19-05-2020**	**Name:**	**BINDUSHRI**
Course:	**TCSion**	**USN:**	**4AL17EC011**
Topic:	**Gain guidance from career gurus Write a winning resume and cover letter. Stay ahead in grp discussion.**	**Semester & Section:**	**6th A**
Github Repository:	**Bindushri**		

FORENOON SESSION DETAILS


g41.tcsion.com/LX/contents/content_home?content_player=true&org_id=1016&TargetOrgid=3876&usrorgid=1016&LaunchFrom=iHUB&User-Agent=Computer&c_id=career-edge-knockdown-the-lockdo...

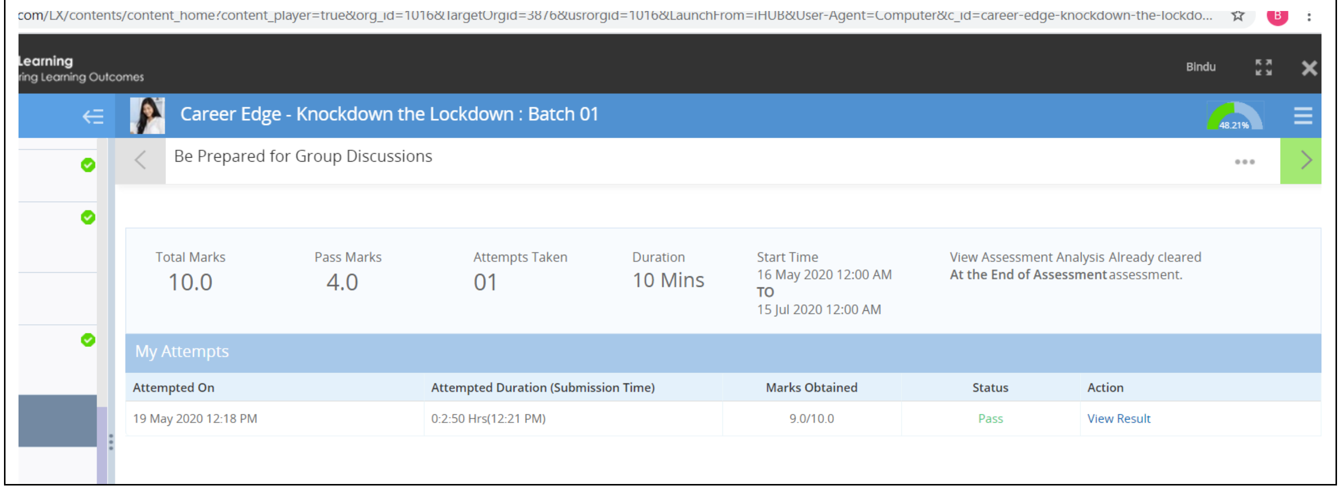
Career Edge - Knockdown the Lockdown : Batch 01

Prepare a Strong Resume and Cover Letter

Total Marks	Pass Marks	Attempts Taken	Duration	Start Time	View Assessment Analysis
10.0	4.0	01	10 Mins	16 May 2020 12:00 AM TO 15 Jul 2020 12:00 AM	Already cleared At the End of Assessment assessment.

My Attempts

Attempted On	Attempted Duration (Submission Time)	Marks Obtained	Status	Action
19 May 2020 11:44 AM	0:2:47 Hrs(11:46 AM)	8.0/10.0	Pass	View Result


com/LX/contents/content_home?content_player=true&org_id=1016&TargetOrgid=3876&usrorgid=1016&LaunchFrom=iHUB&User-Agent=Computer&c_id=career-edge-knockdown-the-lockdo...

Career Edge - Knockdown the Lockdown : Batch 01

Be Prepared for Group Discussions

Total Marks	Pass Marks	Attempts Taken	Duration	Start Time	View Assessment Analysis
10.0	4.0	01	10 Mins	16 May 2020 12:00 AM TO 15 Jul 2020 12:00 AM	Already cleared At the End of Assessment assessment.

My Attempts

Attempted On	Attempted Duration (Submission Time)	Marks Obtained	Status	Action
19 May 2020 12:18 PM	0:2:50 Hrs(12:21 PM)	9.0/10.0	Pass	View Result

→* why do we need a Headstart?
* Intense complettion
* Talent acquisition
* employable skills
* changing job roles
* Employment outlook-positive.

→ key pillars to get a Headstart
- clarity of thought
- Access and reliability
- early preparation
- Acquire Relevant skills.
- Compelling Resume
- cracking the interview

Q

a) write a winning resume and cover letter.

* Resume has power to get an interview for your dreamjob.
* Same may Resume has the power to hold c your dreamjob and remain as dream itself.

Structure of Resume contains:-

1) Contact Details.
2) objective
3) education
4) Skills
5) Personal Detail

Types of a Resume
1) Chronological Resume 2) functional 3) combination

Resume's is not about including everything But it is all about including write thing.
* cover letter gives the initial impression.

Structure of a cover letter

~~Date~~
~~Your name~~
~~Your~~

Summary

→ the resume should be crisp & to the Point
→ the resume should be clear about your career objective, skill, abilities and what your looking for
→ donot lie in the resume.
→ cover letter gives initial impression

3) Stay ahead in the group Discussion

* group-discussion is a part of interview process
* group-discussion is a true exchang of a views on a particular topic
* it is conducted to check a interpersonal skill,

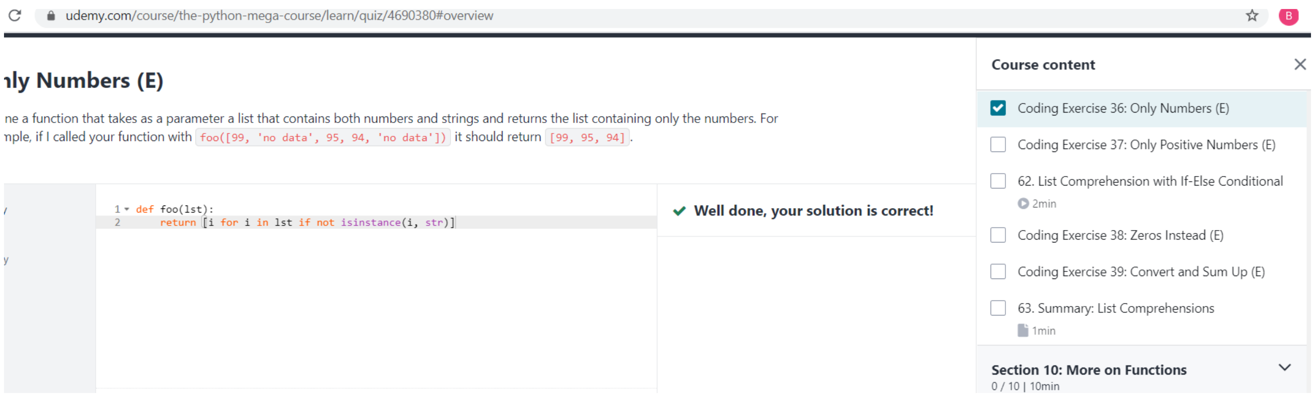
points to remember
* clarity * Bodylanguage * listening
* Tone of voice * Appropriate language.
* courtesy * conciseness * correctness

Summary
* group discussions not a debate
* To be aware of your Body language
* To keep a check on your tone of voice & the language used.

Date:19may2020	**Name:Bindushri**
Course: python	**USN:4AL17EC011**
Topic: Basics:sec	**Sem&Sec:6th A**

AFTERNOON SESSION DETAILS

Image of session



udemy.com/course/the-python-mega-course/learn/quiz/4690380#overview

ıly Numbers (E)

ne a function that takes as a parameter a list that contains both numbers and strings and returns the list containing only the numbers. For
ıple, if I called your function with `foo([99, 'no data', 95, 94, 'no data'])` it should return `[99, 95, 94]`.

```
def foo(lst):
    return [i for i in lst if not isinstance(i, str)]
```

✓ Well done, your solution is correct!

Course content

- ✓ Coding Exercise 36: Only Numbers (E)
- Coding Exercise 37: Only Positive Numbers (E)
- 62. List Comprehension with If-Else Conditional
  2min
- Coding Exercise 38: Zeros Instead (E)
- Coding Exercise 39: Convert and Sum Up (E)
- 63. Summary: List Comprehensions
  1min

Section 10: More on Functions
0 / 10 | 10min

Section 8:- putting the pieces Together

* Problem statement
* To split a big task into smaller task
* Inorder to that first need to create the function.

ex:-

```
1.   def sentence_maker(phrase):
2.       capitalized = phrase.capitalize()
#this will look like
    >>> "how are you".capitalize()
    >>    'how are you'
    >> "how are you". startswith(("how", "what","why"))
        True.
    >> "its a good weather".startswith(("how", "what", "why"))
        False.
```

ex:-

```
1.   def sentence_maker(phrase):
2.   interrogatives = ("how", "what", "why")
3.   capitalized = phrase. capitalize()
4.   if phrase. starts with (interrogatives):
5.       return "{}?". format (capitalized)
6. else:
7.       return "{}.". format (capitalized)
8.
9. print (sentence_maker ("how are you"))
```

o/p
how are you?

```
# for constructing the loop
10. results = []
11. while True:
12.         user_input = input ("Say something:")
```

```
13. if user_input == "\end":
14.     break
15. else:
        results.append (sentence_maker (user_input))
16.
17
18. print (results)   -> O/P -> say something: howareyou
                                say something: weather is good
```

~~* To concatenate all string into one st~~

```
                                say something: /end.
                        -> ['howareyou', 'weather is good']
```

* To concatenate a string into one single string then make use of.

```
>>> "-".join(["how are you", "good good"])
>>> 'how are you-good good'
```

```
# In the same code.
18. print (" ".join (results))
```

O/P ->
```
saysomething: how are you
saysomething: weather is good
saysomething: \end.
how are you . weather is good.
```

Section 9:- List comprehensions

```
temp = [221, 234, 340, 230]
new_temps = []
for temp in temps:
        new_temps.append (temp /10)
print (new_temps)
```

List ~~compression is nothing but er~~
```
temps = [221, 234, 340, 230]
newtemps = [temp/10 for temp in temps]
print(new_temps)
```

o/p

22.1, 23.4, 34.0, 23.0.

This is the list comprehension ~~where~~ ~~no need of using for loop~~

* List comprehension with if conditional

```
temps = [221, 224, 340, -9999, 230]
newtemps = [temp/10 for temp in temps if
                        temp != -9999]
print (new_temps)
```

o/p ⇒ 22.1, 22.4, 34.0, 23.0

Summary: - list comprehensions
• A list comprehension is an expression that creates a list by iterating over another container
• Basic list comprehension
   - [i*2 for i in [1, 5, 10]]
   o/2 -> 2, 10, 20

Section 10 :-

Function with Multiple Arguments

```
def area (a,b)
    return a*b
print (area (4,5))
```

O/P → 20

function with an arbitrary number of non-keyword argument

```
1. def mean (*args):
       return sum(args) / len (args)

print (mean(1, x=3, 4))
```

function with argument

```
def mean (**kwargs):
    return kwargs
print(mean (a=1, b=2 , c=3))
```

O/P
```
{'a':1 , 'b':2 , 'c':3}
```

Section 11:- File processing

→ Note: there should be a 2 editor windows
1. .py 2. .txt in order to create the file object.

```
1. Myfile = open ("fruits.txt")
2. print (myfile.read())
```

* write some content in the .txt page
ex:- 1. pear
2. apple
3. orang.

* nowu execute program in .py page
the o/p will be.

o/p :- Pear.
apple
orange.

→ Closing a file

```
myfile = open("fruits.txt")
content = myfile.read()
my.file.close()
Print (content)
```

o/p :- pear
apple
orange.

→ Opening the file ~~with~~ using "with"

```
myfile = open ("fruits.txt")
content = myfile.read()
myfile.close()
with open ("fruits.txt") as my file:
    Content = my file.read()

Print (content)
```

→ writing text to a file.

```
with open ("files/fruits.txt", "w") as myfile:
```

classmate
Date
Page

```
content = myfile.write()
~~print (concept)~~
print (concept)
```

read to read the file use "r"
to write on the file use "w".

Section 12:- Imported Modules

```
while true:
        with open("files/vegetables.txt") as file:
            print(file.read())
```

o/p

```
Tomato
  :
  :
Tomato.
```

```
1. import time
2.
3. while True:
4.     with open("files/vegetables.txt") as file:
         print (file.read()
         time.sleep (10)
```

o/p:- # what ever written in the text file that will be executed due to importing module.

Standard Python Module

```
import temo
import os
whlle true:
        if os.path.exists ("fles/vegetables.text"):
         with open ("files/vegetables.txt") as file:
           print (file.read())
  else:
        print ("file does not exist.")
  time. sleep (10).
```

Summary

Bulltin objects are all objects that are written inside the python enterpreter in C language.
Bulltin modules contains Bullt ino objects