Training Day- 7

Unpacking in Python is a way to assign the elements of an iterable (like a list, tuple, or dictionary) to multiple variables in a single operation. Python uses unpacking to simplify and make the code cleaner. Here's a detailed explanation

Python support Unpacking directly

```
"""
# a,b,c,d,e="CETPA"
# print(a)
# print(b)
# print(c)
# print(d)
# print(e)
……..New Program…….
# a,b,c,d="CAT"       #ValueError
# print(a)
# print(b)
# print(c)
# print(d)
```

Whenever we call a print function then it prints the space separated arguments on the screen ie space is automatically in between the arguments, in parallel after printing all the arguments, a new line in printed ie cursor automatically moves to next line after print statement is executed.
space in print function is printed on the screen automatically if there are more than 1 arguments, space is made to separate the arguments on the screen.

```
……..New Program…….
# a,b,c,d,e=2,3,4,5,6
# print(a,b,c)
# print(d,e)
……..New Program…….
# print()
# print()
# print()
```

Escape Characters in Python: These characters leaves behind a special

functionality on the screen but in actual these characters are not printed itself on the screen \t: Tab Character
\n: New Line Character

```
……..New Program…….
# s="CE\tTP\tA"
# print(s)
……..New Program…….
# s="CE   TP  A"
# print(s)

……..New Program…….
# s="CE\nt\n\nPA"
# print(s)

# #New Program
# s="CETPA"
# print(s)

# #New Program
# s='CETPA'
# print(s)

# #New Program
# s='"CETPA"'
# print(s)

# #New Program
# s='"'CETPA'"'
# print(s)
```

Optional Parameters In Print Statement: 'end' and 'sep'

Default value of end = '\n'
Default of sep = ' ' ie space
We can consider end and sep as variables in python. When we print our arguments then in between the arguments automatically sep is printed and at the end of the arguments end is printed.

```
……..New Program…….
# a,b,c,d,e=1,2,3,4,5
# print(a,b,c)        #asepbsepcend  print(a,sep,b,sep,c,end)
# print(d,e)          #dsepeend    print(d,sep,e,end)

……..New Program…….
```

```
# a,b,c,d,e=1,2,3,4,5
# print(a,b,c,sep="*",end="$")
# print(d,e,end="@")
```

SANDBOX: Development Environment

PRODUCTION ENVIRONMENT: Live Environment publish

```
# #New Program
# s="ce't'p'a"
# print(s)
```

……..New Program…….

```
# a,b,c,d,e=1,2,3,4,5
# print(a,end=" ")       #end space
# print(b,end=" ")
# print(c,end=" ")
# print(d,end=" ")
# print(e,end="")     #end empty string
```

……..New Program…….

```
# a,b,c,d,e=1,2,3,4,5
# print(a,end="")       #end empty string
# print(b,end="")
# print(c,end="")
# print(d,end="")
# print(e,end="")     #end empty string
```

```
# #New Program
# a,b,c,d,e=1,2,3,4,5
# print(a,b,c,d,e,sep="")
```

……..New Program…….

```
# print("\tCETPA")
# print("C\tCETPA")
# print("CE\tCETPA")
# print("CET\tCETPA")
# print("CETP\tCETPA")
# print("CETPA\tCETPA")
```

Variable: Data Storing Element whose value varies in the program.

How variables are created in Python: By assigning the value

"""

```
# a=b     #NameError: name 'b' is not defined
# print(a)

……..New Program…….
# a=5
# print(a)
```

Predefined Names: Keywords

User Defined Names: Identifiers

Examples of Identifiers: Variable Names, Function Names, Class Names Rules To Define Identifiers In Python: Possible valid identifiers:

1. All English Alphabets Are Allowed Ie Upper Case And Lower Case
2. Numbers Are Allowed From 0 To 9 But Identifier Name Should Not Start With Numbers.

Numbers Can Be Used In Between Or End Of The Identifiers.

3. Special Symbol: Only Underscore Ie _ Is Allowed 4. Can't Have Any Special Symbol Other Than Underscore
5. Can't Have Any Keyword Names.

```
"""
# a b=5         #Not allowed space
# print(a b)
# cetpa=7       #Allowed
# print(cetpa)
# a*b=9         #Not allowed *
# print(a*b)
# true=5        #Allowed
# print(true)
# True=5        #Keywords not allowed
# print(True)
# _a=5          #Allowed
# print(_a)
# __a__="CETPA" #Allowed
# print(__a__)
# 5a="CETPA"      #Not allowed, can't start with numbers
# print(5a)
# a5=100      #Allowed
# print(a5)

# #New Program
# _5a=6
# print(_5a)

"""
```

Assignment:

8. If i = 3, j = 2 what is the result of following expressions?

a. i + 5 >= j – 6

b. j * 10 < i ** 2

c. i < j + 5 > j ** 4

"""

#New Program
i = 3
j = 2
print(i + 5 >= j - 6) #8>-4
print(j * 10 < i ** 2) #20 <9
print(i < j + 5 > j ** 4) # 3<7 >16

#New Program
7. If the radius of a circle is 3 meters, find the area of the circle.
r=3
a=3.14*r*r
print(a)