Sakina Jaorawala

Python Intermediate

Lesson 1

Full code

```
print('Welcome to grade 7')
```

Explanation

The print() function is used to display the specified message on the screen.

Code

```
print
```

Explanation

print is the keyword

Your notes here

Debugging Lesson 2

Instruction:

Debugging print and Syntax

Code

```
Welcome to grade 7
```

Lesson 3

Full code

```
a='hello'
print(a)
b=3
print(b)
c=5.6
print(c)
```

Explanation

Variables are like containers for storing information in a computer program.

Code

```
a='hello'
print(a)
b=3
print(b)
c=5.6
print(c)
```

Explanation

a, b and c are variables

Your notes here

Debugging Lesson 4

Instruction:

Debugging variables and Syntax

Code

```
$a='hello'
print(a)
2b=3
print(b)
8c=5.6
print(c)
```

Lesson 5

Full code

```
_12a=2
print(_12a)
a12=2
print(a12)
```

Explanation

Variable names must begin with letters or the underscore (_) symbol.

Code

```
_12a=2
print(_12a)
a12=2
print(a12)
```

Explanation

Variables may include uppercase and lowercase letters from A to Z, numbers from 0 to 9, and special characters.

Your notes here

Debugging Lesson 6

Instruction:

Debugging naming rules and Syntax

Code

```
__12a=2
print(_12a)
a12=2
print(a12)
```

Lesson 7

Full code

```
a=10
print(a)
A=11
print(A)
```

Code

```
a=10
print(a)
A=11
print(A)
```

Explanation

Variables are case sensitive

Your notes here

Debugging Lesson 8

Instruction:

Debugging naming rules and Syntax

Code

```
a==10
print(a)
A==11
print(A)
```

Lesson 9

Full code

```
a='hello'
print(a)
```

```
b="hello"
print(b)
c='''hi
     how are you'''
print(c)
d="""hi
   how are you"""
print(d)
```

Explanation

You can use either double quotes or single quotes to enclose strings.

Code

```
a='hello'
print(a)
b="hello"
print(b)
c='''hi
     how are you'''
print(c)
d="""hi
   how are you"""
print(d)
```

Explanation

For multi-line strings, you should use either triple single quotes or triple double quotes to enclose them.

Your notes here

Debugging Lesson 10

Instruction:

Debugging string creation and Syntax

Code

```
a='hello'
print(a)
b="hello"
print(b)
c=''hi
     how are you''
print(c)
d=""hi
   how are you""
print(d)
```

Lesson 11

Full code

```
a=12
print(a)
b=-12
print(b)
c=12.34
print(c)
d=-12.34
print(d)
```

Explanation

An integer can be either a positive or a negative value.

Code

```
a=12
print(a)
b=-12
print(b)
c=12.34
```

Explanation

A floating-point number can be positive or negative and includes decimal values.

Your notes here

Debugging Lesson 12

Instruction:

Debugging integer and Syntax

Code

```
a=12
print(a)
b=*12
print(b)
c=12.34
print(c)
d=-12.34
print(d)
```

Lesson 13

Full code

```
a=True
print(a)
b=False
print(b)
```

Explanation

Boolean values will be True and False

Your notes here

Debugging Lesson 14

Instruction:

Debugging boolean values and Syntax

Code

```
a=true
print(a)
b=false
print(b)
```

Lesson 15

Full code

```
x=int(1)
print(x)
y=int(2.8)
print(y)
z=int("3")
print(z)
```

Explanation

The process of changing one data type into another is called type casting.

Code

```
x=int(1)
print(x)
y=int(2.8)
print(y)
z=int("3")
print(z)
```

Explanation

Type casting to integer.

Your notes here

Debugging Lesson 16

Instruction:

Debugging Type casting and Syntax

Code

```
x=integer(1)
print(x)
y=integer(2.8)
print(y)
z=integer("3")
print(z)
```

Lesson 17

Full code

```
x=float(1)
print(x)
y=float(2.8)
print(y)
z=float("3")
print(z)
w=float("4.2")
print(w)
```

Explanation

Type casting to float.

Your notes here

Debugging Lesson 18

Instruction:

Debugging Type casting and Syntax

Code

```
x=Float(1)
print(x)
y=Float(2.8)
print(y)
z=Float("3")
print(z)
w=Float("4.2")
print(w)
```

Lesson 19

Full code

```
x=str("s1")
y=str(2)
z=str(3.0)
print(x,y,z)
```

Explanation

Type casting to string.

Your notes here

Debugging Lesson 20

Instruction:

Debugging Type casting and Syntax

Code

```
x=string("s1")
y=string(2)
z=string(3.0)
print(x,y,z)
```

Lesson 21

Full code

```
a=9
b=6
print(a+b)
print(a-b)
print(a*b)
print(a/b)
print(a**b)
print(a%b)
print(a//b)
```

Explanation

Arithmetic operations.

Code

```
a=9
b=6
print(a+b)
```

Explanation

Addition.

Code

```
a=9
b=6
print(a+b)
print(a-b)
```

Explanation

Subtraction.

Code

```
a=9
b=6
print(a+b)
print(a-b)
print(a*b)
```

Explanation

Multiplication.

Code

```
a=9
b=6
print(a+b)
print(a-b)
print(a*b)
print(a/b)
```

Explanation

Division.

Code

```
a=9
b=6
print(a+b)
print(a-b)
print(a*b)
print(a/b)
print(a**b)
```

Explanation

Exponentiation or power.

Code

```
a=9
b=6
print(a+b)
print(a-b)
print(a*b)
print(a/b)
print(a**b)
print(a%b)
```

Explanation

Modulus or remainder.

Code

```
a=9
b=6
print(a+b)
print(a-b)
```

```
print(a*b)
print(a/b)
print(a**b)
print(a%b)
print(a//b)
```

Explanation

Floor division.

Your notes here

Debugging Lesson 22

Instruction:

Debugging Addition, subtraction and multiplication, division

Code

```
a=9
b=6
print(a+b)
print(a-b)
print(a*b)
print(a/b)
print(a**b)
print(a%b)
print(a\b)
```

Lesson 23

Full code

```
apple=50
orange=60
if apple>orange:
    print('Apples are costlier')
else:
    print('Oranges are costlier')
```

Code

```
apple=50
orange=60
if apple>orange:
```

Explanation

If the cost of apples is higher, then it will print "Apples are costlier."

Explanation

If conditions

Code

```
apple=50
orange=60
if apple>orange:
    print('Apples are costlier')
else:
```

Explanation

If the cost of oranges is higher, then the else part will be executed.

Your notes here

Debugging Lesson 24

Instruction:

Debugging If statement, and Syntax

Code

```
apple=50
orange=60
if apple>orange
print('Apples are costlier')
else
```

```
print('Oranges are costlier')
```

Lesson 25

Full code

```
a=10
b=50
c=4
if a>b & a>c:
    print('a is greater than b and c')
else:
    print('a is not greater than b and c')
```

Explanation

Multiple conditions in If

Code

```
a=10
b=50
c=4
if a>b & a>c:
```

Explanation

With AND, both conditions must be met, while with OR, at least one of the conditions should be satisfied.

Your notes here

Debugging Lesson 26

Instruction:

Debugging and, or operator and Syntax

Code

```
a=10
b=50
c=4
if a>b && a>c:
    print('a is greater than b and c')
else:
    print('a is not greater than b and c')
```

Lesson 27

Full code

```
num1=int(input('Enter a number'))
num2=int(input('Enter another number'))
print(num1+num2)
```

Explanation

Getting input from user.

Code

```
num1=int(input('Enter a number'))
num2=int(input('Enter another number'))
print(num1+num2)
```

Explanation

Take two numbers as input, then add them together.

Your notes here

Debugging Lesson 28

Instruction:

Debugging Input and Syntax

Code

```
num1=int(Input'Enter a number')
num2=int(Input'Enter another number')
print(num1+num2)
```

Lesson 29

Full code

```
num1,num2=input().split()
print(int(num1)+int(num2))
```

Code

```
num1,num2=input().split()
print(int(num1)+int(num2))
```

Explanation

Convert strings into numbers, and then perform addition.

Code

```
num1,num2=input().split()
```

Explanation

Getting multiple inputs using split

Your notes here

Debugging Lesson 30

Instruction:

Debugging split and Syntax

Code

```
num1,num2=input.split
print(int(num1)+int(num2))
```

Lesson 31

Full code

```
i=1
while i<=5:
    print('Hello')
    i=i+1
```

Explanation

While loop

Code

```
i=1
while i<=5:
```

Explanation

We use loops to run a set of instructions repeatedly.

Code

```
i=1
while i<=5:
    print('Hello')
    i=i+1
```

Explanation

The while loop will keep running the code block as long as the value of i is less than or equal to 5.

Your notes here

Debugging Lesson 32

Instruction:

Debugging while and Syntax

Code

```
i=1
```

```
while i<=5
print('Hello')
i=i+1
```

Lesson 33

Full code

```
n=5
while n>0:
    n=n-1
    if n==2:
        break
    print(n)
print('Loop ended.')
```

Code

```
n=5
while n>0:
```

Explanation

The code will keep executing until the condition is satisfied.

Code

```
n=5
while n>0:
    n=n-1
    if n==2:
        break
```

Explanation

If we use the break statement within the loop, it will stop the loop.

Your notes here

Debugging Lesson 34

Instruction:

Debugging break and Syntax

Code

```
n=5
while n>0:
    n=n-1
if n==2:
    break
    print(n)
print('Loop ended.')
```

Lesson 35

Full code

```
n=5
while n>0:
    n-=1
    if n==2:
        continue
    print(n)
print('Loop ended.')
```

Code

```
n=5
while n>0:
    n-=1
    if n==2:
        continue
```

Explanation

The ongoing iteration ends using the continue command.

Your notes here

Debugging Lesson 36

Instruction:

Debugging list creation and len,append, pop, copy, count, del

Code

```
n=5
while n>0:
    n-=1
    if n==2
    continue
    print(n)
print('Loop ended.')
```

Lesson 37

Full code

```
list=['audi','bmw','benz','nexa']
print(list)
print(len(list))
list.append('Toyota')
print(list)
list.pop()
print(list)
newlist=list.copy()
print(newlist)
print(list.count('bmw'))
del newlist
```

Explanation

A list is a collection of items. A list should always start and end with []. Lists are mutable, meaning you can change their elements (add, remove, or modify) after the listÂ isÂ created.

Code

```
list=['audi','bmw','benz','nexa']
print(list)
print(len(list))
```

Explanation

The len() function determines the size of the list.

Code

```
list=['audi','bmw','benz','nexa']
print(list)
print(len(list))
list.append('Toyota')
```

Explanation

The append() method adds an item to the end of the list.

Code

```
list=['audi','bmw','benz','nexa']
print(list)
print(len(list))
list.append('Toyota')
print(list)
list.pop()
```

Explanation

The pop() method will remove item from the list.

Code

```
list=['audi','bmw','benz','nexa']
print(list)
print(len(list))
list.append('Toyota')
print(list)
list.pop()
print(list)
newlist=list.copy()
```

Explanation

The copy() method will create a copy of the list.

Code

```
list=['audi','bmw','benz','nexa']
print(list)
print(len(list))
list.append('Toyota')
print(list)
list.pop()
```

```
print(list)
newlist=list.copy()
print(newlist)
print(list.count('bmw'))
```

Explanation

The count() method calculates how many times an element appears in the list

Code

```
list=['audi','bmw','benz','nexa']
print(list)
print(len(list))
list.append('Toyota')
print(list)
list.pop()
print(list)
newlist=list.copy()
print(newlist)
print(list.count('bmw'))
del newlist
```

Explanation

The del() function deletes the entire list.

Your notes here

Debugging Lesson 38

Instruction:

Debugging list creation and len,append, pop, copy, count, del

Code

```
list=['audi','bmw','benz','nexa']
print(list)
print(length(list))
list.add('Toyota')
```

```
print(list)
list.pop()
print(list)
newlist=list.copy()
print(newlist)
print(list.count('bmw'))
del newlist
```

Lesson 39

Full code

```
a=('red','blue','green')
print(a)
print(len(a))
print(a.count('red'))
del a
```

Explanation

Tuple should start and end with (). Tuples are immutable, meaning their elements cannot be changed after creation.

Code

```
a=('red','blue','green')
print(a)
print(len(a))
```

Explanation

The len() function determines the size of the tuple.

Code

```
a=('red','blue','green')
print(a)
print(len(a))
print(a.count('red'))
```

Explanation

The count() method calculates how many times an element appears in the tuple

Code

```
a=('red','blue','green')
```

```
print(a)
print(len(a))
print(a.count('red'))
del a
```

Explanation

The del() function deletes the entire tuple.

Your notes here

Debugging Lesson 40

Instruction:

Debugging tuple creation and len,count,del

Code

```
a=('red','blue','green')
print(a)
print(len(a))
print(a.count('red'))
delete a
```

Lesson 41

Full code

```
a={'red','blue','green'}
print(a)
a.add('yellow')
print(a)
b={'violet','pink'}
a.update(b)
print(a)
a.remove('green')
print(a)
a.clear()
print(a)
```

```
del a
```

Explanation

Define a data type.

Code

```
a={'red','blue','green'}
```

Explanation

Creating a set. Sets should be surrounded by {}.

Code

```
a={'red','blue','green'}
print(a)
a.add('yellow')
```

Explanation

Add an element to a set

Code

```
a={'red','blue','green'}
print(a)
a.add('yellow')
print(a)
b={'violet','pink'}
a.update(b)
```

Explanation

Update set a with set b

Code

```
a={'red','blue','green'}
print(a)
a.add('yellow')
print(a)
b={'violet','pink'}
a.update(b)
print(a)
a.remove('green')
```

Explanation

Removes an element from a set.

Code

```
a={'red','blue','green'}
print(a)
a.add('yellow')
print(a)
b={'violet','pink'}
a.update(b)
print(a)
a.remove('green')
print(a)
a.clear()
```

Explanation

Clears all elements from the set.

Code

```
a={'red','blue','green'}
print(a)
a.add('yellow')
print(a)
b={'violet','pink'}
a.update(b)
print(a)
a.remove('green')
print(a)
a.clear()
print(a)
del a
```

Explanation

Deletes the set.

Your notes here

Debugging Lesson 42

Instruction:

Debugging set methods and Syntax

Code

```
a={'red','blue','green'}
print(a)
a.add{'yellow'}
print(a)
b={'violet','pink'}
a.update{b}
print(a)
a.remove{'green'}
print(a)
a.clear()
print(a)
del a
```

Lesson 43

Full code

```
dictionary={'car1':'maruti','car2':'verna','car3':'innova'}
print(dictionary)
print(len(dictionary))
print(dictionary.get('car1'))
dictionary.update({'seater':4})
print(dictionary)
dictionary.pop('car2')
print(dictionary)
print(dictionary.items())
print(dictionary.keys())
print(dictionary.values())
newdict=dictionary.copy()
print(newdict)
newdict.clear()
print(newdict)
del newdict
```

Explanation

Dictionary datatype. In the dictionary, each element has a key and valueÂ pair. Dictionary should be

surrounded by {}.

Code

```
dictionary={'car1':'maruti','car2':'verna','car3':'innova'}
print(dictionary)
print(len(dictionary))
```

Explanation

The len() function counts how many key-value pairs are there.

Code

```
dictionary={'car1':'maruti','car2':'verna','car3':'innova'}
print(dictionary)
print(len(dictionary))
print(dictionary.get('car1'))
```

Explanation

The get() method returns the key's value.

Code

```
dictionary={'car1':'maruti','car2':'verna','car3':'innova'}
print(dictionary)
print(len(dictionary))
print(dictionary.get('car1'))
dictionary.update({'seater':4})
```

Explanation

The update() method inserts the specified key and value into the dictionary.

Code

```
dictionary={'car1':'maruti','car2':'verna','car3':'innova'}
print(dictionary)
print(len(dictionary))
print(dictionary.get('car1'))
dictionary.update({'seater':4})
print(dictionary)
dictionary.pop('car2')
```

Explanation

The pop() method will delete the key and value pair.

Code

```
dictionary={'car1':'maruti','car2':'verna','car3':'innova'}
print(dictionary)
print(len(dictionary))
print(dictionary.get('car1'))
dictionary.update({'seater':4})
print(dictionary)
dictionary.pop('car2')
print(dictionary)
print(dictionary.items())
print(dictionary.keys())
print(dictionary.values())
```

Explanation

The corresponding output is obtained through the use of items, keys, and values.

Your notes here

Debugging Lesson 44

Instruction:

Debugging dictionary methods and Syntax

Code

```
dictionary={'car1':'maruti','car2':'verna','car3':'innova'}
print(dictionary)
print(length(dictionary))
print(dictionary.get('car1'))
dictionary.update(('seater':4))
print(dictionary)
dictionary.pop('car2')
print(dictionary)
print(dictionary.items())
print(dictionary.keys())
print(dictionary.values())
newdict=dictionary.copy{}
print(newdict)
```

```
newdict.clear()
print(newdict)
del newdict
```

Lesson 45

Full code

```
str='hello how are you'
print(str.upper())
print(str.lower())
print(str.capitalize())
print(str.count('h'))
print(str.isalnum())
print(str.isdigit())
print(str.split())
```

Code

```
str='hello how are you'
print(str.upper())
```

Explanation

upper() method converts string to upper case.

Code

```
str='hello how are you'
print(str.upper())
print(str.lower())
```

Explanation

lower() method converts string to lower case.

Code

```
str='hello how are you'
print(str.upper())
print(str.lower())
print(str.capitalize())
```

Explanation

capitalize() method converts the first character of a string to uppercase and the rest to lowercase.

Code

```
str='hello how are you'
print(str.upper())
print(str.lower())
print(str.capitalize())
print(str.count('h'))
```

Explanation

count() method is used to count the number of occurrences of a substring in a given string.

Code

```
str='hello how are you'
print(str.upper())
print(str.lower())
print(str.capitalize())
print(str.count('h'))
print(str.isalnum())
```

Explanation

isalnum() method checks if all the characters in a given string are alphanumeric (consisting of only letters and numbers) and returns True if so, and False otherwise.

Code

```
str='hello how are you'
print(str.upper())
print(str.lower())
print(str.capitalize())
print(str.count('h'))
print(str.isalnum())
print(str.isdigit())
```

Explanation

isdigit() method checks if all the characters in a given string are digits (numeric characters from 0 to 9). It returns True if all characters are digits and False otherwise.

Code

```
str='hello how are you'
print(str.upper())
print(str.lower())
print(str.capitalize())
print(str.count('h'))
print(str.isalnum())
print(str.isdigit())
```

```
print(str.split())
```

Explanation

split() method is used to split a string into a list of substrings based on a specified delimiter or separator. By default, the delimiter is a space character.

Your notes here

Debugging Lesson 46

Instruction:

Debugging string methods and Syntax

Code

```
str='hello how are you'
print(str.upper())
print(str.lower())
print(str.capitalize())
print(str.count('h'))
print(str.isAlnum())
print(str.isdigit())
print(str.issplit())
```

Lesson 47

Full code

```
num=[5,4,3,2,1]
res=0
for x in num:
    res=res+x
    print('value of x',x)
print('Total is',res)
```

Explanation

For loop will iterate through an iterable object.

Code

```
num=[5,4,3,2,1]
res=0
for x in num:
```

Explanation

Iterating through the list.

Your notes here

Debugging Lesson 48

Instruction:

Debugging for loop and addition operation

Code

```
num=[5,4,3,2,1]
res=0
for x in num
res=res+x
    print('value of x',x)
print('Total is',res)
```

Lesson 49

Full code

```
for x in range(5):
    print(x)
else:
    print('successful')
```

Code

```
for x in range(5):
```

Explanation

range() function is used to generate a sequence of numbers within a specified range.(0 to 4 in this example)

Code

```
for x in range(5):
    print(x)
else:
    print('successful')
```

Explanation

Else part will be executed if for loop is successfully executed.

Your notes here

Lesson 51

Full code

```
age=40
print('senior citizen') if age>=60 else print('not a senior citizen')
```

Explanation

Ternary operators

Code

```
age=40
print('senior citizen') if age>=60 else print('not a senior citizen')
```

Explanation

When the condition is met, the true condition will be executed; otherwise, the false condition will be carried out.

Your notes here

Lesson 52

Full code

```
a,b=10,20
min=a if a<b else b
print(min)
```

Explanation

Conditional expressions are also known as ternary operators.

Code

```
a,b=10,20
```

Explanation

a will be assigned with a value of 10 and b will be assigned with 20

Code

```
a,b=10,20
min=a if a<b else b
```

Explanation

If the condition is true, a will be stored in min; otherwise, b will be stored.

Your notes here

Debugging Lesson 53

Instruction:

Debugging ternary operator and if else

Code

```
a,b=10,20
min=a
if a<b else b
print(min)
```

Quiz 54

Question

The evaluation of ternary operators is based on conditions or expressions. Is the given statement True or False?

Options

True

False

The ternary operator take three arguments: The first is a comparison argument. The second is the result upon a true comparison. The third is the result upon a false comparison. [true_condition] if [expression] else [false_condition]

Lesson 55

Full code

```
a,b=10,20
print((b, a) [a<b])
```

Explanation

Using tuple to choose an item

Code

```
a,b=10,20
print((b, a) [a<b])
```

Explanation

syntax :(if_test_false,if_test_true)[test]

Your notes here

Lesson 56

Full code

```
a,b=10,20
print({True:a, False:b} [a<b])
```

Explanation

Using Dictionary for choosing an item

Code

```
a,b=10,20
print({True:a, False:b} [a<b])
```

Explanation

if [a < b] is true, the value of the True key will be printed; otherwise, the value of the False key will be printed.

Your notes here

Quiz 58

Question

What is ternary operator in python?

Options

The ternary operator in python is used to return a value based on the result of a binary condition.

The ternary operator in python is used to return a value based on the result of a ternary operation.

The ternary operator in python is used to return a value based on the result of a binary condition. It takes binary value(condition) as an input, so it looks similar to an "if-else"

condition block. However, it also returns a value so behaving similar to a function.

Test 59

Write a Python code that takes an individual's age as input and prints their age category based on the following criteria: If the age is 60 or above, print 'Senior Citizen.' If the age is between 18 and 59 (inclusive), print 'Adult.' If the age is below 18, print 'Minor.'

Write solution here

Lesson 60

Full code

```
age=15
if age>12:
    if age<20:
        print('Hurray, You are in teenage')
```

Explanation

If you complete your schoolwork and have dinner on time, I will take you along for shopping can be an example of Nested loop.

Code

```
age=15
if age>12:
```

Explanation

Age should be greater then 12.

Code

```
age=15
if age>12:
    if age<20:
```

Explanation

Age should be less than 20.

Code

```
age=15
if age>12:
    if age<20:
        print('Hurray, You are in teenage')
```

Explanation

The print statement will be executed if both conditions are satisfied.

Your notes here

Quiz 61

Question

```
num = 15
num = 15
if num >= 0:
  if num == 0:
    print("Zero")
  else:
  print("Positive number")
else:
  print("Negative number")
```

What will be the output of the following code?

Options

```
Positive number

negative  number
```

The condition will check whether the number is positive or negative or zero.

Quiz 62

Question

```
x = 41
if x > 10:
  print("Above ten,")
  if x > 20:
    print("and also above 20!")
  else:
    print("but not above 20.")
```

```
 What will be the output of the above code?
```

Options

```
Above ten,
and also above 20!

Below ten,
and also below 20!
```

The condition verifies whether the number is greater than 10 and greater than 20. If both conditions are satisfied, the loop will be broken and the statement will be carried out. The else portion will be carried out if the condition is false.

Debugging Lesson 63

Instruction:

Debugging nested if and > < operarors

Code

```
age = 15
if age>12:
if age<20:
    print('Hurray, You are in teenage')
```

Lesson 64

Full code

```
num=int(input('enter number'))
if num>=0:
    if num==0:
        print("zero")
    else:
        print("positive number")
else:
    print("negative number")
```

Explanation

Multiple If conditions

Code

```
num=int(input('enter number'))
if num>=0:
```

Explanation

The outer if statement checks if num is greater than or equal to 0.

Code

```
num=int(input('enter number'))
if num>=0:
    if num==0:
```

Explanation

If true, it enters the inner if-else block.

- If num is equal to 0, it prints "zero."

- If num is greater than 0, it prints "positive number."

Code

```
num=int(input('enter number'))
if num>=0:
    if num==0:
        print("zero")
    else:
        print("positive number")
else:
```

Explanation

If the outer if condition is false, it goes to the else block and prints "negative number."

Your notes here

Lesson 65

Full code

```
num = int(input('enter number'))
if num > 0:
    print("positive number")
elif num == 0:
    print("zero")
else:
    print("negative number")
```

Explanation

Using if, elif, and else.

Code

```
num = int(input('enter number'))
if num > 0:
    print("positive number")
```

Explanation

print "positive number" if the value of num is greater than 0.

Code

```
num = int(input('enter number'))
if num > 0:
    print("positive number")
elif num == 0:
    print("zero")
```

Explanation

This elif block handles the case where num is not greater than 0 but is equal to 0. If both the if and elif

conditions are false, it would move to the else block

Code

```
num = int(input('enter number'))
if num > 0:
    print("positive number")
elif num == 0:
    print("zero")
else:
    print("negative number")
```

Explanation

It prints "negative number."

Your notes here

Debugging Lesson 66

Instruction:

Debugging nested if..elif...else and print

Code

```
num=int(input('enter number'))
if num>0:
    print("positive number")
else if num==0:
    print("zero")
else:
    print("negative number")
```

Quiz 67

Question

`elif is a combination of _____.`

Options

Two if statements

else and if statements

The elif is short for else if. It allows us to check for multiple expressions. If the condition for if is False , it checks the condition of the next elif block and so on. If all the conditions are False , the body of else is executed.

Lesson 68

Full code

```
marks=int(input('enter marks'))
if marks<=40:
    print('grade D')
elif marks>40 and marks<=60:
    print('grade C')
elif marks>60 and marks<=80:
    print('grade B')
else:
    print('grade A')
```

Explanation

Program for finding grades

Code

```
marks=int(input('enter marks'))
```

Explanation

Multiple elif and if conditions can be used

Code

```
marks=int(input('enter marks'))
if marks<=40:
    print('grade D')
elif marks>40 and marks<=60:
```

Explanation

And operator is used to check multiple conditions in one If statement

Your notes here

Lesson 69

Full code

```
marks=int(input('enter marks'))
if marks<=40:
    print('grade D')
if marks>40 and marks<=60:
    print('grade C')
if marks>60 and marks<=80:
    print('grade B')
if marks>80:
    print('grade A')
```

Code

```
marks=int(input('enter marks'))
if marks<=40:
```

Explanation

Different If conditions for different marks ranges.

Code

```
marks=int(input('enter marks'))
if marks<=40:
    print('grade D')
if marks>40 and marks<=60:
    print('grade C')
if marks>60 and marks<=80:
    print('grade B')
if marks>80:
```

Explanation

No else is used, all ranges are handled using If conditions.

Your notes here

Debugging Lesson 70

Instruction:

Debugging nested if and > < operarors

Code

```
marks=int(input('enter marks'))
if marks=<40:
    print('grade D')
if marks>40 and marks=<60:
    print('grade C')
if marks>60 and marks=<80:
    print('grade B')
if marks>80:
    print('grade A')
```

Quiz 71

Question

`Is it possible to use elif and if together, for example, elif if 6>7: ?`

Options

`Yes`

`No`

It is not possible to use elif and if together. The elif is short for else if. It allows us to check for multiple expressions. If the condition for if is False , it checks the condition of the next elif block and so on. If all the conditions are False , the body of else is executed.

Lesson 72

Full code

```
year = int(input('Enter year: '))
if (year % 4 == 0 and year % 100 != 0) or (year % 400 == 0):
    print("Leap Year")
elif year < 0:
```

```
    print("Invalid Input: Please enter a positive year")
else:
    print("Non Leap Year")
```

Explanation

Get year as user input.

Code

```
year = int(input('Enter year: '))
if (year % 4 == 0 and year % 100 != 0) or (year % 400 == 0):
    print("Leap Year")
```

Explanation

The first if block checks for a leap year using the specified conditions.

Code

```
year = int(input('Enter year: '))
if (year % 4 == 0 and year % 100 != 0) or (year % 400 == 0):
    print("Leap Year")
elif year < 0:
    print("Invalid Input: Please enter a positive year")
```

Explanation

The elif block checks if the year is less than 0, indicating an invalid input.

Code

```
year = int(input('Enter year: '))
if (year % 4 == 0 and year % 100 != 0) or (year % 400 == 0):
    print("Leap Year")
elif year < 0:
    print("Invalid Input: Please enter a positive year")
else:
    print("Non Leap Year")
```

Explanation

The else block handles the case of a non-leap year when none of the previous conditions are met.

Your notes here

Debugging Lesson 73

Instruction:

Debugging nested if...elif...else and and or operators

Code

```
year = str(input('Enter year: '))
if (year % 4 == 0 and year % 100 != 0) or (year % 400 == 0):
      print("Leap Year")
elif year < 0:
    print("Invalid Input: Please enter a positive year" )
else:
    print( " Non Leap Year " )
```

Quiz 74

Question

Is there a limit to use number of if statements that can be used in a programme?

Options

Yes

no

There is no virtual limit in nesting if-else statements.

Lesson 75

Full code

```
a=50
b=10
c=15
if a>b:
    if a>c:
        print('a is greater')
if b>a:
    if b>c:
```

```
        print('b is greater')
if c>a:
    if c>b:
        print('c is greater')
```

Explanation

Find a greater number among three numbers

Code

```
a=50
b=10
c=15
if a>b:
    if a>c:
```

Explanation

Condition is checked if a is greater than b and c.

Code

```
a=50
b=10
c=15
if a>b:
    if a>c:
        print('a is greater')
if b>a:
    if b>c:
```

Explanation

Condition is checked if b is greater than a and c.

Code

```
a=50
b=10
c=15
if a>b:
    if a>c:
        print('a is greater')
if b>a:
    if b>c:
```

```
        print('b is greater')
if c>a:
    if c>b:
```

Explanation

Condition is checked if c is greater than a and b.

Your notes here

Lesson 77

Full code

```
import turtle
turtle.showturtle();
turtle.shape('turtle');
color=input('enter color')
size=input('enter size')
if color in ['red','green','blue','yellow','pink','black']:
    if size=='big':
        turtle.pencolor(color)
        turtle.circle(95)
    if size=='small':
        turtle.pencolor(color)
        turtle.circle(35)
```

Code

```
import turtle
turtle.showturtle();
turtle.shape('turtle');
```

Explanation

Select shape of the turtle

Code

```
import turtle
turtle.showturtle();
```

```
turtle.shape('turtle');
color=input('enter color')
```

Explanation

Get colour as input

Code

```
import turtle
turtle.showturtle();
turtle.shape('turtle');
color=input('enter color')
size=input('enter size')
```

Explanation

Get size as input

Code

```
import turtle
turtle.showturtle();
turtle.shape('turtle');
color=input('enter color')
size=input('enter size')
if color in ['red','green','blue','yellow','pink','black']:
```

Explanation

Check if colour is present in list

Your notes here

Lesson 79

Full code

```
fruits='oranges'
if fruits=='apples':
    buy='apples'
elif fruits=='oranges':
```

```
    buy='oranges'
else:
    buy='grapes'
print(buy)
```

Code

```
fruits='oranges'
if fruits=='apples':
```

Explanation

If you like apples,then buy apples

Code

```
fruits='oranges'
if fruits=='apples':
    buy='apples'
elif fruits=='oranges':
    buy='oranges'
```

Explanation

Else if you like oranges, then buy oranges

Code

```
fruits='oranges'
if fruits=='apples':
    buy='apples'
elif fruits=='oranges':
    buy='oranges'
else:
    buy='grapes'
```

Explanation

If you buy grapes instead, it indicates that you do not enjoy apples or oranges.

Your notes here

Quiz 80

Question

```
i = 1
while i < 0:
  print(i)
  i += 1
else:
  print("i is no longer less than 6")
```

What is the output?

Options

`i is no longer less than 6`

`1 2 3 4 5 6`

Since In the given code, the value of i is less than 0, the condition becomes false and the else part will get executed.

Quiz 81

Question

Combination of elif is _____.

Options

`else & else statements`

`else & if statements`

The elif is a combination of if and else statements. If the condition is true it will print the statement otherwise the else part will get executed.

Test 82

Write a Python code that prompts the user to enter a number. If the number is even and greater than 10, print "Even and greater than 10." If the number is odd, print "Odd." If the number is less than or equal to 10, print "Less than or equal to 10."

Write solution here

Lesson 83

Full code

```
print("Hello", "welcome to grade7",'happy learning')
print("Hello", "welcome to grade7",'happy learning',sep='@')
print("Hello", "welcome to grade7",'
happy learning')
print("Hello", "welcome to grade7",'happy learning',end='@')
```

Explanation

Advanced print statements.

Code

```
print("Hello", "welcome to grade7",'happy learning')
```

Explanation

A print statement can display multiple strings simultaneously.

Code

```
print("Hello", "welcome to grade7",'happy learning')
print("Hello", "welcome to grade7",'happy learning',sep='@')
```

Explanation

@' is separator, it specifies how to separate the objects

Code

```
print("Hello", "welcome to grade7",'happy learning')
print("Hello", "welcome to grade7",'happy learning',sep='@')
```

```
print("Hello", "welcome to grade7",'
happy learning')
```

Explanation

Will print "happy learning" in new line

Code

```
print("Hello", "welcome to grade7",'happy learning')
print("Hello", "welcome to grade7",'happy learning',sep='@')
print("Hello", "welcome to grade7",'
happy learning')
print("Hello", "welcome to grade7",'happy learning',end='@')
```

Explanation

End specifies what to print at the end

Your notes here

Debugging Lesson 84

Instruction:

Debugging print and separator,end,new line

Code

```
print("Hello", "welcome to grade7",'happy learning')
print("Hello", "welcome to grade7",'happy learning',seperator='@')
print("Hello", "welcome to grade7",'
happy learning')
print("Hello", "welcome to grade7",'happy learning',end='@')
```

Lesson 85

Full code

```
age=input('enter your age')
print('i am ',age,' years old')
```

Code

```
age=input('enter your age')
print('i am ',age,' years old')
```

Explanation

Print can have a combination of strings and variables

Your notes here

Quiz 86

Question

What is end argument used for in print statement?

Options

To print something at end

To separate strings

The end parameter in the print function is used to add any string. At the end of the output of the print statement in python. By default, the print function ends with a newline.

Lesson 87

Full code

```
amount=65
txt='amount paid is {} rupees'
print(txt.format(amount))
products=3
txt='amount {} rupees is paid for {} products'
print(txt.format(amount,products))
```

Explanation

Print formatting

Code

```
amount=65
txt='amount paid is {
```

Explanation

Creating a placeholder where amount has to be displayed.

Code

```
amount=65
txt='amount paid is {} rupees'
print(txt.format(amount))
```

Explanation

The format() method will pass the amount to the placeholder.

Code

```
amount=65
txt='amount paid is {} rupees'
print(txt.format(amount))
products=3
txt='amount {} rupees is paid for {} products'
print(txt.format(amount,products))
```

Explanation

Two values are passed to the placeholder.

Your notes here

Lesson 88

Full code

```
amount=65
products=3
print(f' amount {amount} rupees is paid for {products} products')
print(f' amount {amount:.2f} rupees is paid for {products} products')
```

Code

```
amount=65
products=3
print(f
```

Explanation

The 'f' is also used for print formatting.

Code

```
amount=65
products=3
print(f' amount {amount}
```

Explanation

amount value will get placed here.

Code

```
amount=65
products=3
print(f' amount {amount} rupees is paid for {products} products')
print(f' amount {amount:.2f
```

Explanation

2f will place 2 decimal values after the amount.

Your notes here

Quiz 90

Question

Format and f are used for _____.

Options

Print formatting

Loop formatting

Also called "formatted string literals", f-strings are string literals that have an f at the beginning and curly braces containing expressions that will be replaced with their values. The expressions are evaluated at runtime and then formatted using the __format__ protocol.

Test 91

Write a Python code that prompts the user to enter their name and their favorite color. Using an f-string, print a message in the format "Hello [Name]! Your favorite color is [Color]."

Write solution here

Lesson 92

Full code

```
mylist=['dog','cat','swam','fly','cow']
print(mylist[0])
print(mylist[1])
print(mylist[3])
```

Explanation

Indexing in list

Code

```
mylist=['dog','cat'
```

Explanation

Indexes begin at 0. In this case, the dog will be indexed as 0 and the cat as 1.

Code

```
mylist=['dog','cat','swam','fly','cow'
```

Explanation

Swam will have index value as 2, fly will have index value as 3, cow will have index value as 4

Code

```
mylist=['dog','cat','swam','fly','cow']
print(mylist[0])
```

Explanation

mylist[0] will get the element at index 0 i.e dog in our list

Code

```
mylist=['dog','cat','swam','fly','cow']
print(mylist[0])
print(mylist[1])
```

Explanation

The cat is at index 1

Code

```
mylist=['dog','cat','swam','fly','cow']
print(mylist[0])
print(mylist[1])
print(mylist[3])
```

Explanation

The fly is at index 3

Your notes here

Lesson 93

Full code

```
mylist=['dog','cat','swam','fly','cow']
print(mylist[0:2])
print(mylist[1:3])
print(mylist[2:4])
```

Explanation

Index slicing

Code

```
mylist=['dog','cat','swam','fly','cow']
print(mylist[0
```

Explanation

Specify the Starting position

Code

```
mylist=['dog','cat','swam','fly','cow']
print(mylist[0:2
```

Explanation

Specify the end position, till the 2nd index (excluded).

Code

```
mylist=['dog','cat','swam','fly','cow']
print(mylist[0:2])
print(mylist[1:3])
```

Explanation

Slicing starts from index 1 to index 3. here index 3 will be excluded so till index 2 , it will display.

Your notes here

Debugging Lesson 94

Instruction:

Debugging list and indexing

Code

```
mylist=['dog','cat','swam','fly','cow']
print(mylist(0))
print(mylist(1))
print(mylist(3))
```

Debugging Lesson 95

Instruction:

Debugging list and indexing,slicing

Code

```
mylist={'dog','cat','swam','fly','cow'}
print(mylist[0:2])
print(mylist[1:3])
print(mylist[2:4])
```

Quiz 96

Question

```
mylist=['dog','cat','swam','fly','cow']
mylist[1:4]

What will be the output?
```

Options

```
['dog','cat','swam','fly']
```

```
['cat','swam','fly']
```

We have use the concept of slicing wherein first and fourth element will be eliminated and remaining elements will get printed.

Lesson 97

Full code

```
mylist=['dog','cat','swam','fly','cow']
print(mylist[0:])
print(mylist[:3])
```

```
print(mylist[:])
```

Explanation

Default index positions

Code

```
mylist=['dog','cat','swam','fly','cow']
print(mylist[0
```

Explanation

Index value starts from 0.

Code

```
mylist=['dog','cat','swam','fly','cow']
print(mylist[0:
```

Explanation

Stop index is not specified so it will take default which is the last element of list.

Code

```
mylist=['dog','cat','swam','fly','cow']
print(mylist[0:])
print(mylist[
```

Explanation

Since start index is not specified, index 0 will be used by default.

Code

```
mylist=['dog','cat','swam','fly','cow']
print(mylist[0:])
print(mylist[:3])
print(mylist[:])
```

Explanation

Both indexes are not specified so start index default will be 0 and stop index default will be last element of the list.

Your notes here

Debugging Lesson 98

Instruction:

Debugging index and default values

Code

```
mylist=['dog','cat','swam','fly','cow']
print(mylist{0:})
print(mylist{:3})
print(mylist{:})
```

Lesson 99

Full code

```
mylist=[34,56,77,89]
print(mylist[0:])
print(mylist[:3])
print(mylist[:])
```

Explanation

Index slicing with numbers.

Code

```
mylist=[34,56,77,89]
print(mylist[0:])
```

Explanation

Default stop index will be last element of list.

Code

```
mylist=[34,56,77,89]
print(mylist[0:])
print(mylist[:3])
```

Explanation

Default start index will be 0.

Code

```
mylist=[34,56,77,89]
print(mylist[0:])
print(mylist[:3])
print(mylist[:])
```

Explanation

Both indexes will be default values.

Your notes here

Quiz 100

Question

```
mylist=[34,56,77,89]
print(mylist[:])

What will be the output?
```

Options

Lesson 101

Full code

```
mylist=['dog','cat','swam','fly','cow','rat','frog']
print(mylist[1:7])
print(mylist[1:7:2])
print(mylist[1:7:3])
print(mylist[1:7:4])
```

Code

```
mylist=['dog','cat','swam','fly','cow','rat','frog']
print(mylist[1:7])
```

Explanation

Get elements from index 1 to index 7 (7 excluded)

Code

```
mylist=['dog','cat','swam','fly','cow','rat','frog']
print(mylist[1:7])
print(mylist[1:7:2])
```

Explanation

It will skip one element between because a skip size of 2 is specified.

Code

```
mylist=['dog','cat','swam','fly','cow','rat','frog']
print(mylist[1:7])
print(mylist[1:7:2])
print(mylist[1:7:3])
```

Explanation

It will skip two elements between because a skip size of 3 is specified.

Code

```
mylist=['dog','cat','swam','fly','cow','rat','frog']
print(mylist[1:7])
print(mylist[1:7:2])
print(mylist[1:7:3])
print(mylist[1:7:4])
```

Explanation

It will skip three elements between because a skip size of 4 is specified.

Your notes here

Debugging Lesson 102

Instruction:

Debugging index slicing and skip size

Code

```
mylist=['dog','cat','swam','fly','cow','rat','frog']
print(mylist[8])
print(mylist[1:7:2])
print(mylist[1:7:3])
print(mylist[1:7:4])
```

Quiz 103

Question

How do you skip items in Python?

Options

By the break and continue statements

By the loop

The break and continue statements in Python are used to skip parts of the current loop or break out of the loop completely.

Lesson 104

Full code

```
mylist=['dog','cat','swam','fly','cow']
print(mylist[-1])
print(mylist[-2])
print(mylist[-4])
```

Explanation

Negative indexing or backward indexing.

Code

```
mylist=['dog','cat','swam','fly','cow']
print(mylist[-1])
```

Explanation

Backward indexing starts from the last element.

Code

```
mylist=['dog','cat','swam','fly','cow']
print(mylist[-1])
```

```
print(mylist[-2])
```

Explanation

mylist[-2] will give you the element before the last element in the list.

Code

```
mylist=['dog','cat','swam','fly','cow']
print(mylist[-1])
print(mylist[-2])
print(mylist[-4])
```

Explanation

Start counting from last. So, mylist[-4] will give value cow.

Your notes here

Lesson 105

Full code

```
mylist=['dog','cat','swam','fly','cow']
print(mylist[-1:-4])
print(mylist[-1:-4:-1])
print(mylist[-4:-1])
```

Explanation

Index slicing using negative indexing

Code

```
mylist=['dog','cat','swam','fly','cow']
print(mylist[-1:-4])
```

Explanation

The skip size should also be negative because mylist[-1] is the last element and mylist[-4] is the element cat, which is its opposite.

Code

```
mylist=['dog','cat','swam','fly','cow']
print(mylist[-1:-4])
print(mylist[-1:-4:-1])
```

Explanation

skip size of -1 will work

Your notes here

Debugging Lesson 106

Instruction:

Debugging list and negative indexing

Code

```
mylist=['dog','cat','swam','fly','cow']
print(mylist[-1:-4])
print(mylist[-1:-4:-1:2])
print(mylist[-4:-1])
```

Quiz 107

Question

```
mylist=['dog','cat','swam','fly','cow']
print(mylist[-1:-4])

What is the output?
```

Options

```
[Cow,fly,swam ]

[]
```

Python supports "indexing from the end", that is, negative indexing. This means the last value of a sequence has an index of -1, the second last -2, and so on. In the given code skip size of -1 is missing.

Lesson 108

Full code

```
import turtle
colors=['red', 'purple', 'blue', 'green', 'orange', 'yellow']
t=turtle.Turtle()
for x in range(6):
    t.pencolor(colors[x])
    t.forward(50)
    t.left(60)
```

Code

```
import turtle
colors=['red', 'purple', 'blue', 'green', 'orange', 'yellow']
```

Explanation

list of colours is created

Code

```
import turtle
colors=['red', 'purple', 'blue', 'green', 'orange', 'yellow']
t=turtle.Turtle()
for x in range(6):
```

Explanation

Read through the list using for loop

Code

```
import turtle
colors=['red', 'purple', 'blue', 'green', 'orange', 'yellow']
t=turtle.Turtle()
for x in range(6):
    t.pencolor(colors[x])
```

Explanation

List indexing is used to fetch colours one by one.

Your notes here

Lesson 109

Full code

```
import turtle
colors=['red','purple','blue','green','orange','yellow']
t=turtle.Turtle()
for x in range(100):
    t.pencolor(colors[x%6])
    t.width(x//100 + 1)
    t.forward(x)
    t.left(59)
```

Code

```
import turtle
colors=['red','purple','blue','green','orange','yellow']
```

Explanation

List of colour is created.

Code

```
import turtle
colors=['red','purple','blue','green','orange','yellow']
t=turtle.Turtle()
for x in range(100):
```

Explanation

Repeat the loop for 100 times.

Code

```
import turtle
colors=['red','purple','blue','green','orange','yellow']
t=turtle.Turtle()
for x in range(100):
    t.pencolor(colors[x%6])
    t.width(x//100 + 1)
    t.forward(x)
    t.left(59)
```

Explanation

Forms a pattern.

Your notes here

Lesson 111

Full code

```
mytup=('chennai','delhi','cochin','pune')
print(mytup[0])
print(mytup[3])
print(mytup[-1])
print(mytup[-4])
```

Code

```
mytup=('chennai','delhi','cochin','pune')
print(mytup[0])
```

Explanation

Index 0 position has value Chennai.

Code

```
mytup=('chennai','delhi','cochin','pune')
print(mytup[0])
print(mytup[3])
```

Explanation

Index 3 position has value pune.

Code

```
mytup=('chennai','delhi','cochin','pune')
print(mytup[0])
print(mytup[3])
print(mytup[-1])
```

Explanation

In negative indexing, -1 has the last element pune.

Code

```
mytup=('chennai','delhi','cochin','pune')
print(mytup[0])
print(mytup[3])
print(mytup[-1])
print(mytup[-4])
```

Explanation

In negative indexing, -4 has the element chennai.

Your notes here

Lesson 112

Full code

```
mytup=('chess','swimming','foot ball','hockey','basket ball')
print(mytup[0:4])
print(mytup[3:])
print(mytup[-1:-5:-1])
print(mytup[-4::])
```

Code

```
mytup=('chess','swimming','foot ball','hockey','basket ball')
print(mytup[0
```

Explanation

Index start position is 0

Code

```
mytup=('chess','swimming','foot ball','hockey','basket ball')
print(mytup[0:4
```

Explanation

Index stop position is 4(excluded).

Code

```
mytup=('chess','swimming','foot ball','hockey','basket ball')
print(mytup[0:4])
print(mytup[3:
```

Explanation

Index slicing from 3 till end of the tuple.

Code

```
mytup=('chess','swimming','foot ball','hockey','basket ball')
print(mytup[0:4])
print(mytup[3:])
print(mytup[-1:-5:-1])
```

Explanation

Negative slicing from -1 to -5, with skip size of -1.

Code

```
mytup=('chess','swimming','foot ball','hockey','basket ball')
print(mytup[0:4])
print(mytup[3:])
print(mytup[-1:-5:-1])
print(mytup[-4::])
```

Explanation

Prints a subsequence of the tuple starting from the fourth-to-last element ('swimming') and includes all elements up to the end of the tuple.

Your notes here

Debugging Lesson 113

Instruction:

Debugging tuple and indexing,slicing

Code

```
mytup=('chess','swimming','foot ball','hockey','basket ball')
```

```
print([0:4])
print([3:])
print([-1:-5:-1])
print([-4::])
```

Quiz 114

Question

```
mytup=('chess','swimming','foot ball','hockey','basket ball')
print(mytup[-4::])
```

What will be the output?

Options

('swimming', 'foot ball', 'hockey', 'basket ball')

[]

In negative indexing, the last value of a sequence has an index of -1, the second last -2, and so on. In the code, the end value is default so all the elements will be printed by default.

Lesson 115

Full code

```
mytup=('cricket','baseball','hockey','chess','football')
print(mytup[0:3])
print(mytup[3:5])
print(mytup[-1:-4:-1])
print(mytup[-4:-1])
```

Code

```
mytup=('cricket','baseball','hockey','chess','football')
print(mytup[0
```

Explanation

Index start position is 0

Code

```
mytup=('cricket','baseball','hockey','chess','football')
print(mytup[0:3
```

Explanation

Index stop position is 3 (excluded)

Code

```
mytup=('cricket','baseball','hockey','chess','football')
print(mytup[0:3])
print(mytup[3:5])
print(mytup[-1
```

Explanation

Negative indexing start position -1.

Code

```
mytup=('cricket','baseball','hockey','chess','football')
print(mytup[0:3])
print(mytup[3:5])
print(mytup[-1:-4
```

Explanation

Negative indexing stop position -4 (excluded).

Code

```
mytup=('cricket','baseball','hockey','chess','football')
print(mytup[0:3])
print(mytup[3:5])
print(mytup[-1:-4:-1
```

Explanation

Reverse direction so skip size is -1.

Your notes here

Debugging Lesson 116

Instruction:

Debugging tuple and negative indexing, sclicing

Code

```
mytup=('cricket','baseball','hockey','chess','football')
print(mytup[0:3])
print(mytup[3:5:1:1])
print(mytup[-1:-4:-1])
print(mytup[-4:-1])
```

Lesson 117

Full code

```
mydict={'car':'Innova','model':'K123','colour':'white','seater':4}
print(mydict['car'])
print(mydict['colour'])
```

Explanation

Indexing in dictionary

Code

```
mydict={'car':'Innova','model':'K123','colour':'white','seater':4}
print(mydict['car'])
```

Explanation

In dictionaries, keys are used for indexing.

Code

```
mydict={'car':'Innova','model':'K123','colour':'white','seater':4}
print(mydict['car'])
print(mydict['colour'])
```

Explanation

Colour key will give the value.

Your notes here

Lesson 118

Full code

```
newdict={'name1':'vishal','name2':'sam','name3':'anu'}
for x,y in enumerate(newdict):
    print(x,y)
```

Code

```
newdict={'name1':'vishal','name2':'sam','name3':'anu'}
for x,y in enumerate
```

Explanation

The enumerate() function is used to iterate over a sequence (such as a list, tuple, or string) along with its index. It returns pairs of the form (index, element).

Your notes here

Quiz 120

Question

`Dict[key] will get _____.`

Options

`Keys`

`Values`

Python's efficient key/value hash table structure is called a "dict". The contents of a dict can be written as a series of key:value pairs within braces { }, e.g. dict = {key1:value1, key2:value2, ... }. dict[key] will return its value.

Test 121

Create a list fruits with three fruit names.Print the second fruit using indexing and a subset with the first and third fruits using slicing. Create a tuple numbers with three integers and print the last number using indexing. Create a dictionary subjects_dict with subject names and corresponding grades, print the first grade using key indexing, and use enumerate to print each subject with its position.

Write solution here

Lesson 122

Full code

```
i=1
while i<=3:
    print(i*'@')
    j=1
    while j<=2:
        print(j*'#')
        j=j+1
    i=i+1;
```

Explanation

Nested while loops

Code

```
i=1
```

Explanation

Loop inside a loop is called nested loops.

Code

```
i=1
```

```
while i<=3:
```

Explanation

Outer while loop.

Code

```
i=1
while i<=3:
    print(i*'@')
```

Explanation

prints @ 1 time

Code

```
i=1
while i<=3:
    print(i*'@')
    j=1
    while j<=2:
```

Explanation

Inner while loop

Code

```
i=1
while i<=3:
    print(i*'@')
    j=1
    while j<=2:
        print(j*'#')
```

Explanation

Inner while loop executes once and twice.

Code

Explanation

Inner looping is completed before the control goes to outer loop.

Your notes here

Debugging Lesson 123

Instruction:

Debugging nested while loop and pattern creation

Code

```
i=1
while i<=3:
    print(i*'@')
    j=1
while j<=2:
        print(j*'#')
        j=j+1
    i=i+1;
```

Lesson 124

Full code

```
i=1
while i<=3:
    j=1
    while j<=4:
        print(i,j)
        j=j+1
    i=i+1
```

Code

```
i=1
```

Explanation

Initialize the outer loop variable.

Code

```
i=1
while i<=3:
```

Explanation

Loop should go on till the value of i is 3.

Code

```
i=1
while i<=3:
    j=1
```

Explanation

Initialize the inner loop variable.

Code

```
i=1
while i<=3:
    j=1
    while j<=4:
```

Explanation

Loop should go on till the value of j is 4.

Code

```
i=1
while i<=3:
    j=1
    while j<=4:
        print(i,j)
```

Explanation

For value i=1, the inner loop gets executed when j= 1,2,3,4.

Your notes here

Quiz 125

Question

```
i = 1
```

```
while i < 6:
  print(i)
  if i == 5:
    break
  i += 1
```

what will be the output?

Options

1 2 3 4 5

1 2 3 4 5 6

The statement will print the first 5 numbers.

Lesson 126

Full code

```
i=1
while i<=3:
    j=1
    print(i,' pattern')
    while j<=4:
        print(j*'*')
        j=j+1
    i=i+1
```

Code

```
i=1
while i<=3:
```

Explanation

Outer loop is for printing the pattern 3 times

Code

```
i=1
while i<=3:
    j=1
    print(i,' pattern')
    while j<=4:
```

Explanation

Inner loop is for printing the stars - 1 star, 2 star, 3 star and 4 stars.

Your notes here

Debugging Lesson 127

Instruction:

Debugging nested while loop and pattern creation

Code

```
i=1
while i<=3:
    j=1
    print(i,' pattern')
  while j<=4:
        print(j + '*')
        j=j+1
    i=i+1
```

Lesson 128

Full code

```
i=1
while i<=3:
    j=1
    while j<=5:
        print(i,"*",j,'=',i*j)
        j=j+1
    i=i+1
    print('
')
```

Explanation

Let's print multiplication tables.

Code

```
i=1
while i<=3:
```

Explanation

Outer loop to print till 3 tables.

Code

```
i=1
while i<=3:
  j=1
  while j<=5:
```

Explanation

Inner loop to print 5 rows in each table.

Code

```
i=1
while i<=3:
  j=1
  while j<=5:
        print(i,"*",j,'=',i*j)
```

Explanation

Multiplying i and j will give results.

Your notes here

Debugging Lesson 129

Instruction:

Debugging nested while and multiplication table

Code

```
i=1
while i<=3:
```

```
  j=1
  while j<=5
        print(i "*" j '=',i*j)
        j=j+1
  i=i+1
  print('
')
```

Quiz 130

Question

```
adj = ["red", "big"]
fruits = ["apple"]

for x in adj:
  for y in fruits:
    print(x, y)
```

what will be the output?

Options

```
red apple
big apple
```

```
apple
big red apple
```

A nested loop is a loop inside a loop. The "inner loop" will be executed one time for each iteration of the "outer loop".

Lesson 131

Full code

```
i=5
while i>=3:
    print(i*'%')
    j=3
    while j>=1:
        print(j*'$')
        j=j-1
    i=i-1
```

Explanation

Decrementing the variable in a while loop.

Code

```
i=5
```

Explanation

Initially 5 % has to be printed so initialize the variable with 5.

Code

```
i=5
while i>=3:
    print(i*'%')
```

Explanation

Multiply i value with string %

Code

```
i=5
while i>=3:
```

Explanation

Outer loop will get executed till i is >=3.

Code

```
i=5
while i>=3:
    print(i*'%')
    j=3
    while j>=1:
        print(j*'$')
        j=j-1
    i=i-1
```

Explanation

Decrementing the i value means reducing 1 from the i value.

Your notes here

Debugging Lesson 132

Instruction:

Debugging nested while and Syntax

Code

```
i=5
While i>=3:
    print(i*'%')
    j=3
    While j>=1:
        print(j*'$')
        j=j-1
    i=i-1
```

Lesson 133

Full code

```
i=5
while i>=1:
    j=3
    while j>=1:
        print('i value:',i,'  j value:',j)
        j=j-1
    i=i-1
    print('
')
```

Explanation

Number pattern printing

Code

```
i=5
while i>=1:
```

Explanation

Execute numbers in reverse from 5 to 1.

Code

```
i=5
while i>=1:
    j=3
    while j>=1:
```

Explanation

Execute numbers in reverse from 3 to 1.

Code

```
i=5
while i>=1:
    j=3
    while j>=1:
        print('i value:',i,'  j value:',j)
```

Explanation

Print the values for i and j and examine the pattern.

Code

```
i=5
while i>=1:
    j=3
    while j>=1:
        print('i value:',i,'  j value:',j)
        j=j-1
    i=i-1
```

Explanation

Decrement i and j so that the numbers will execute in reverse.

Your notes here

Lesson 134

Full code

```
i=1
while i<=5:
    print(i*'* ')
    i=i+1
```

Explanation

Print a triangle

Code

```
i=1
while i<=5:
```

Explanation

The loop should go on until i <=5.

Code

```
i=1
while i<=5:
    print(i*'* ')
```

Explanation

*Multiplying i value with * forms a triangle.*

Your notes here

Lesson 135

Full code

```
i=1
while i<=5:
    print(i*'& ')
```

```
    i=i+1
j=5
while j>=1:
    print(j*'& ')
    j=j-1
```

Explanation

Print a pattern using two while loops.

Code

```
i=1
while i<=5:
    print(i*'& ')
    i=i+1
```

Explanation

Prints first 5 rows of the pattern.

Code

```
i=1
while i<=5:
    print(i*'& ')
    i=i+1
j=5
while j>=1:
```

Explanation

Print the same pattern in reverse.

Your notes here

Debugging Lesson 136

Instruction:

Debugging nested while and pattern forming

Code

```
i=
while i<=5
    print(i*'& ')
    i=i+1
j=5
while j>=1
    print(j*'& ')
    j=j-1
```

Quiz 137

Question

What is nested while loop in Python?

Options

while loop inside another while loop

a while loop after a while loop

When a while loop is present inside another while loop then it is called nested while loop.

Lesson 138

Full code

```
j=5
while j>=1:
    print(j*'* ')
    j=j-1
```

Explanation

Printing triangle in reverse

Code

```
j=5
```

Explanation

Start with printing 5 stars

Code

```
j=5
while j>=1:
    print(j*'* ')
    j=j-1
```

Explanation

Continue decrementing 1 star from 5 stars till it reaches 1 star.

Your notes here

Lesson 139

Full code

```
playernumber=1
while playernumber<=11:
    numberoftries=1
    while numberoftries<=3:

print('playernumber:',playernumber,'numberoftries:',numberoftries)
        numberoftries=numberoftries+1
    playernumber=playernumber+1
```

Explanation

11 players with 3 trials each.

Code

```
playernumber=1
while playernumber<=11:
```

Explanation

Count the players.

Code

```
playernumber=1
while playernumber<=11:
    numberoftries=1
    while numberoftries<=3:
```

Explanation

Count the number of trials.

Your notes here

Lesson 141

Full code

```
import turtle
t=turtle.Turtle()
t.color('black')
i=1
while i<=4:
    t.width(i)
    j=1
    while j<=4:
        t.forward(100)
        t.left(90)
        j=j+1
    i=i+1
```

Explanation

Draw square with different line thickness.

Code

```
import turtle
t=turtle.Turtle()
t.color('black')
i=1
while i<=4:
```

Explanation

Outer loop to repeat square 4 times.

Code

```
import turtle
t=turtle.Turtle()
t.color('black')
i=1
while i<=4:
    t.width(i)
```

Explanation

Width sets different line thickness.

Code

```
import turtle
t=turtle.Turtle()
t.color('black')
i=1
while i<=4:
    t.width(i)
    j=1
    while j<=4:
```

Explanation

Inner loop to draw square with 4 sides.

Your notes here

Lesson 143

Full code

```
import turtle
t=turtle.Turtle()
t.color('black')
i=1
```

```
fwd=[25,50,75]
while i<=4:
    t.width(i)
    j=1
    while j<=4:
        for k in fwd:
            t.forward(k)
            t.left(50)
        j=j+1
    i=i+1
```

Explanation

Nested while.

Code

```
import turtle
t=turtle.Turtle()
t.color('black')
i=1
fwd=[25,50,75]
```

Explanation

Specifies the length to move forward.

Code

```
import turtle
t=turtle.Turtle()
t.color('black')
i=1
fwd=[25,50,75]
while i<=4:
    t.width(i)
    j=1
    while j<=4:
        for k in fwd:
```

Explanation

Reads the length from list.

Your notes here

Test 145

Write a Python program using nested while loops to print the multiplication table of 2 up to 5.

Write solution here

Lesson 146

Full code

```
def display():
    print('Defining a function')
    print('Hello world')

display()
```

Explanation

Functions are sets of statements that do certain tasks.

Code

```
def
```

Explanation

Two parts of functions 1. Define a function. and 2. Calling the function.

Code

```
def display
```

Explanation

Function name

Code

```
def display():
```

Explanation

def is the define keyword

Code

```
def display():
    print('Defining a function')
    print('Hello world')

display()
```

Explanation

Calling the function.

Your notes here

Lesson 147

Full code

```
def addfunc():
    a=10
    b=5
    c=a+b
    print(c)
```

```
addfunc()
```

Code

```
def
```

Explanation

Define the function using def keyword.

Code

```
def addfunc():
    a=10
    b=5
    c=a+b
```

Explanation

This function is to add 2 values.

Code

```
def addfunc():
    a=10
    b=5
    c=a+b
    print(c)
```

Explanation

Defining is complete

Code

```
def addfunc():
    a=10
    b=5
    c=a+b
    print(c)

addfunc()
```

Explanation

Now, call the function by using the function name addfunc().

Your notes here

Debugging Lesson 148

Instruction:

Debugging functions and Syntax

Code

```
function addfunc():
    a=10
    b=5
    c=a+b
    print(c)

addfunc()
```

Quiz 149

Question

How is a function defined?

Options

Using def keyword

Using define keyword

A function is a block of code which only runs when it is called. You can pass data, known as parameters, into a function. A function can return data as a result. It is defined using the keyword "def".

Lesson 150

Full code

```
def addfunc(a,b):
    c=a+b
    print(c)

addfunc(3,4)
```

Code

```
def addfunc(a,b
```

Explanation

The variables a and b are referred as arguments or parameters. A parameter is the variable listed inside the parentheses in the function definition. An argument is the value that is sent to the function when it is called.

Code

```
def addfunc(a,b):
    c=a+b
    print(c)

addfunc(
```

Explanation

Values for a and b should be passed when calling the function

Code

```
def addfunc(a,b):
    c=a+b
    print(c)

addfunc(3,4
```

Explanation

Value 3 will be assigned to'a', and Value 4 to 'b'.

Your notes here

Lesson 151

Full code

```
def addfunc(a,b):
    c=a+b
    return c
```

```
result=addfunc(3,4)
print(result)
```

Code

```
def addfunc(a,b):
    c=a+b
    return c
```

Explanation

Return statement will return the output to the called function.

Code

```
def addfunc(a,b):
    c=a+b
    return c

result
```

Explanation

Output can be stored in a variable, here result is used.

Your notes here

Debugging Lesson 152

Instruction:

Debugging functions, arguments and return

Code

```
define addfunc(a,b):
    c=a+b
    return c

result=addfunc(3,4)
print(result)
```

Quiz 153

Question

What are parameters?

Options

Similar to variables

Keywords

A parameter is the variable listed inside the parentheses in the function definition.

Lesson 154

Full code

```
def addfunc(a,b):
    c=a+b
    return c

result=addfunc('Hello','World')
print(result)
```

Code

```
def addfunc(a,b):
    c=a+b
    return c

result=addfunc('Hello','World'
```

Explanation

Any data type values can be passed to parameters.

Code

```
def addfunc(a,b):
    c=a+b
```

Explanation

Addition or concatenation occurs depending on the data type given.

Your notes here

Lesson 155

Full code

```
def greeting(name,msg='Welcome back!'):
    print(name,',',msg)

greeting('Sam')
greeting('Sam','How do you do?')
```

Explanation

There are two specified arguments.

Code

```
def greeting(name,msg='Welcome back!'):
    print(name,',',msg)

greeting('Sam')
```

Explanation

Value for one argument is passed so the other argument msg will take default value.

Code

```
def greeting(name,msg='Welcome back!'):
    print(name,',',msg)

greeting('Sam')
greeting('Sam','How do you do?')
```

Explanation

Value for both arguments have been passed, thus the output will be printed appropriately.

Your notes here

Debugging Lesson 156

Instruction:

Debugging functions and default values

Code

```
def (name,msg='Welcome back!'):
    print(name,',',msg)

greeting('Sam')
greeting('Sam','How do you do?')
```

Quiz 157

Question

Does all the parameters can be passed in the python language?

Options

Yes

No

All parameters (arguments) in the Python language are passed by reference. It means if you change what a parameter refers to within a function, the change also reflects back in the calling function.

Lesson 158

Full code

```
def greeting(name,msg):
    print(name,',',msg)

greeting(name='Sam',msg='How do you do?')
```

Explanation

Keyword arguments.

Code

```
def greeting(name,msg):
    print(name,',',msg)

greeting(name='Sam',msg='How do you do?')
```

Explanation

Passing the value to parameters using variables is called keyword arguments.

Your notes here

Lesson 159

Full code

```
def greeting(*args):
    for name in args:
        print(name)

greeting('Sam','Anu','Shiv','Arun')
greeting('MsDhoni','Virat','Kapil')
```

Explanation

Arbitrary arguments

Code

```
def greeting
```

Explanation

When we don't know how many values will be passed to the function, in that case use arbitrary arguments.

Code

```
def greeting(*
```

Explanation

** specifies that it's an arbitrary argument.*

Code

```
def greeting(*args
```

Explanation

Any number of values can be passed to this arbitrary argument.

Your notes here

Debugging Lesson 160

Instruction:

Debugging functions and arbitary arguments

Code

```
def greeting(args):
    for name in args:
        print(name)

greeting('Sam','Anu','Shiv','Arun')
greeting('MsDhoni','Virat','Kapil')
```

Quiz 161

Question

Does python accepts multiple keyswords arguments?

Options

Yes

No

*Python can accept multiple keyword arguments, better known as **kwargs. It behaves similarly to *args, but stores the arguments in a dictionary instead of tuples.*

Lesson 162

Full code

```
def example(*args):
    for i in args:
        print(i)

example('python','java')
print('#########')
example('english','maths','physics','chemistry')
```

Code

```
def example(*args)
```

Explanation

**args will accept any number of values.*

Code

```
def example(*args):
    for i in args:
        print(i)

example('python','java')
```

Explanation

Passing 2 values.

Code

```
def example(*args):
    for i in args:
        print(i)

example('python','java')
print('#########')
example('english','maths','physics','chemistry')
```

Explanation

Passing 4 values.

Your notes here

Lesson 163

Full code

```
def example(**kwargs):
    print('Subject 2 is ',kwargs['sub2'])

example(sub1='english',sub2='maths',sub3='physics',sub4='chemistry')
```

Explanation

kwargs stands for key word arguments.

Code

```
def example(**
```

Explanation

*** should be used for kwargs.*

Code

```
def example(**kwargs):
    print('Subject 2 is ',kwargs['sub2'
```

Explanation

Indexing is used here with key as indexes.

Code

```
def example(**kwargs):
    print('Subject 2 is ',kwargs['sub2'])

example(sub1='english'
```

Explanation

Pass values with keys.

Your notes here

Debugging Lesson 164

Instruction:

Debugging functions and keyword arguments

Code

```
def example(kwargs):
    print('Subject 2 is ',kwargs['sub2'])

example(sub1='english',sub2='maths',sub3='physics',sub4='chemistry')
```

Quiz 165

Question

How many values can be passed in arguments in a function?

Options

Any

1.0

Any number of arguments can be passed on to a function according to the requirement.

Lesson 166

Full code

```
import turtle

wn = turtle.Screen()
wn.bgcolor("light green")

t = turtle.Turtle()
t.color("blue")

def sqrfunc(size):
    i=1
    while i<=size:
        t.forward(size)
        t.left(90)
```

```
        size=size-1
        i=i+1

sqrfunc(50)
```

Explanation

Function to draw squares.

Code

```
import turtle

wn = turtle.Screen()
wn.bgcolor("light green")

t = turtle.Turtle()
t.color("blue")

def sqrfunc(size):
```

Explanation

define a function which accepts size value.

Code

```
import turtle

wn = turtle.Screen()
wn.bgcolor("light green")

t = turtle.Turtle()
t.color("blue")

def sqrfunc(size):
    i=1
    while i<=size:
        t.forward(size)
        t.left(90)
        size=size+5
        i=i+1

sqrfunc(2)
```

Explanation

call the function with size value.

Your notes here

Quiz 168

Question

`What does turtle's bgcolor mean?`

Options

`Background color`

`Cursor color`

bg stands for background color. bgcolor("color") is used to give 'color' to the background.

Test 169

Write a function named is_even that takes an integer as a parameter and print True if it's an even number, and False otherwise.

Write solution here

Lesson 170

Full code

```
size=['big','small']
vehicle=['car','van','bus']
for i in size:
    for j in vehicle:
        print(i,j)
```

Explanation

Nested for loops. Outer loop is for size list and inner loop is for vehicle list.

Code

```
size=['big','small']
vehicle=['car','van','bus']
```

Explanation

Two lists named size and vehicle are created.

Code

```
size=['big','small']
vehicle=['car','van','bus']
for i in size:
```

Explanation

Read through size list using for loop.

Code

```
size=['big','small']
vehicle=['car','van','bus']
for i in size:
    for j in vehicle:
```

Explanation

Read through vehicle list using for loop.

Code

```
size=['big','small']
vehicle=['car','van','bus']
for i in size:
```

```
    for j in vehicle:
        print(i,j)
```

Explanation

For each value in size list, inner for loop will be executed 3 times.

Your notes here

Lesson 171

Full code

```
for i in [1,2,3,4]:
    for j in [1,2,3,4]:
        print(i*'*')
```

Code

```
for i in [1,2,3,4]:
```

Explanation

Outer loop will repeat pattern 4 times

Code

```
for i in [1,2,3,4]:
    for j in [1,2,3,4]:
```

Explanation

*Inner loop will repeatedly print *.*

Your notes here

Debugging Lesson 172

Instruction:

Debugging nested for loop and list

Code

```
for i in [1,2,3,4]:
for j in [1,2,3,4]:
        print(i*'*')
```

Quiz 173

Question

`5*"*"  - what will be the output of this?`

Options

`*****`

`5*`

'' is a string here so 5 * '*' will become 5 times * which is *****

Lesson 174

Full code

```
for i in [1,2,3,4]:
    for j in [1,2,3,4]:
        print(j*'*')
```

Code

```
for i in [1,2,3,4]:
```

Explanation

Outer loop will repeat pattern 4 times.

Code

```
for i in [1,2,3,4]:
    for j in [1,2,3,4]:
```

Explanation

*Inner loop will repeat * once, twice, thrice and 4 times.*

Your notes here

Lesson 175

Full code

```
for i in [1,2,3,4]:
    for j in range(1,i+1):
        print("* ",end='')
    print('
')
```

Code

```
for i in [1,2,3,4]:
```

Explanation

Outer loop will print 4 lines.

Code

```
for i in [1,2,3,4]:
    for j in range(1,i+1):
```

Explanation

*Inner loop will print * with a space based on j value.*

Code

```
for i in [1,2,3,4]:
    for j in range(1,i+1):
        print("* ",end='')
    print('
')
```

Explanation

Will begin in a new line.

Your notes here

Debugging Lesson 176

Instruction:

Debugging nested for loop. list and range

Code

```
for i in [1,2,3,4]:
    for j in range(1,i+1):
        print("* ",ending='')
    print(' ')
```

Quiz 177

Question

What is the purpose of in print statement?

Options

Beginning of a new line

End of a new line

Conceptually, moves the cursor to the beginning of the line and then keeps outputting characters as normal.

Lesson 178

Full code

```
for i in [1,2,3,4]:
    for j in [1,2,3,4]:
        if j == i:
            break
        print(i, j)
```

Code

```
for i in [1,2,3,4]:
    for j in [1,2,3,4]:
        if j == i:
```

Explanation

When both i and j are same, the control will break inner loop and goes back to outer loop.

Code

```
for i in [1,2,3,4]:
    for j in [1,2,3,4]:
        if j == i:
            break
```

Explanation

The looping again starts from the outer loop.

Your notes here

Lesson 179

Full code

```
for i in [1,2,3,4]:
    for j in [1,2,3,4]:
        if j == i:
            continue
        print(i, j)
```

Code

```
for i in [1,2,3,4]:
    for j in [1,2,3,4]:
        if j == i:
            continue
```

Explanation

When j value equals i value, the inner loop will skip that iteration and continue reading next value.

Explanation

Nested for with continue statement.

Your notes here

Quiz 181

Question

If there is a break in the inner for loop, will the inner or outer loop be broken?

Options

Inner loop

Outer loop

Because it is inside the inner loop, the inner loop will break.

Lesson 182

Full code

```
import turtle
wn=turtle.Screen()
wn.bgcolor("green")
pen=turtle.Turtle()
pen.color('white')
pen.speed(0)
pen.penup()
pen.goto(-400,100)
for i in range(10):
    pen.write(i)
    pen.right(90)
    pen.forward(10)
    pen.pendown()
    pen.forward(200)
    pen.penup()
    pen.backward(210)
    pen.left(90)
    pen.forward(20)
```

Code

```
import turtle
wn=turtle.Screen()
wn.bgcolor("green")
pen=turtle.Turtle()
pen.color('white')
```

Explanation

Set background colour to green and pen colour to white.

Code

```
import turtle
wn=turtle.Screen()
wn.bgcolor("green")
pen=turtle.Turtle()
pen.color('white')
pen.speed(0)
pen.penup()
pen.goto(-400,100)
for i in range(10):
```

Explanation

Draw 15 lines so range(15) is used.

Code

```
import turtle
wn=turtle.Screen()
wn.bgcolor("green")
pen=turtle.Turtle()
pen.color('white')
pen.speed(0)
pen.penup()
pen.goto(-400,100)
for i in range(10):
    pen.write(i)
```

Explanation

write will print numbers on top.

Your notes here

Lesson 184

Full code

```
import turtle
wn=turtle.Screen()
wn.bgcolor("pink")
pen=turtle.Turtle()
dot_distance = 25
width = 3
height = 5
pen.penup()
for y in range(height):
    for i in range(width):
        pen.dot()
        pen.forward(dot_distance)
    pen.backward(dot_distance * width)
    pen.right(90)
    pen.forward(dot_distance)
    pen.left(90)
```

Explanation

Lets draw dots in sequence.

Code

```
import turtle
wn=turtle.Screen()
wn.bgcolor("pink")
pen=turtle.Turtle()
dot_distance = 25
width = 3
height = 5
pen.penup()
for y in range(height):
```

Explanation

Nested for loops are used to specify height and width.

Code

```
import turtle
wn=turtle.Screen()
wn.bgcolor("pink")
pen=turtle.Turtle()
dot_distance = 25
width = 3
height = 5
pen.penup()
for y in range(height):
    for i in range(width):
```

Explanation

Inner loop will draw 3 dots in each row and outer loop is for number of rows.

Your notes here

Quiz 186

Question

What does the turtle command forward do?

Options

Move the line forward

Move the line backward

Command forward will move the line in forward direction. wheras Command backward will move the line in backward direction.

Test 187

Write a Python program using nested loops to print a right-angled triangle of numbers.

Write solution here

Lesson 188

Full code

```
user_input=str(input("Enter a Phrase: "))
text=user_input.split()
print(text)
a=" "
for i in text:
    a=a+str(i[0]).upper()
print(a)
```

Explanation

Create acronyms using python

Code

```
user_input=str(input("Enter a Phrase: "))
```

Explanation

Get phrase from the user.

Code

```
user_input=str(input("Enter a Phrase: "))
text=user_input.split()
```

Explanation

split() method will split the words and create a list.

Code

```
user_input=str(input("Enter a Phrase: "))
text=user_input.split()
print(text)
a=" "
for i in text:
    a=a+str(i[0]).upper()
```

Explanation

Get the letter at the 0 index position by reading through the list.

Your notes here

Debugging Lesson 189

Instruction:

Debugging input and split, list, upper

Code

```
user_input=int(input("Enter a Phrase: "))
text=user_input.split()
print(text)
a=" "
for i in text:
    a=a+str(i[0]).upper()
print(a)
```

Lesson 190

Full code

```
import random
when=['A few years ago', 'Yesterday', 'Last night', 'A long time
ago','On 20th Jan']
who=['a rabbit', 'an elephant', 'a mouse', 'a turtle','a cat']
name=['Ali', 'Miriam','daniel', 'Hoouk', 'Starwalker']
```

```
place=['Barcelona','India', 'Germany', 'Venice', 'England']
where=['cinema', 'university','seminar', 'school', 'laundry']
occur=['made a lot of friends','Eats a burger', 'found a secret key',
'solved a mistery', 'wrote a book']
print(random.choice(when)+ ', '+random.choice(who)+' that lived in
'+random.choice(place)+', went to the '+random.choice(where)+' and
'+random.choice(occur))
```

Explanation

Story Generator

Code

```
import random
```

Explanation

Random() is a library which will select strings randomly from a sequence.

Code

```
when=['A few years ago', 'Yesterday', 'Last night', 'A long time
ago','On 20th Jan']
who=['a rabbit', 'an elephant', 'a mouse', 'a turtle','a cat']
```

Explanation

Form a list of strings.

Code

```
import random
when=['A few years ago', 'Yesterday', 'Last night', 'A long time
ago','On 20th Jan']
who=['a rabbit', 'an elephant', 'a mouse', 'a turtle','a cat']
name=['Ali', 'Miriam','daniel', 'Hoouk', 'Starwalker']
place=['Barcelona','India', 'Germany', 'Venice', 'England']
where=['cinema', 'university','seminar', 'school', 'laundry']
occur=['made a lot of friends','Eats a burger', 'found a secret key',
'solved a mistery', 'wrote a book']
print(random.choice(when)
```

Explanation

The random.choice() will select string randomly and form a story.

Your notes here

Debugging Lesson 191

Instruction:

Debugging random library and choice

Code

```
when=['A few years ago', 'Yesterday', 'Last night', 'A long time ago','On 20th Jan']
who=['a rabbit', 'an elephant', 'a mouse', 'a turtle','a cat']
name=['Ali', 'Miriam','daniel', 'Hoouk', 'Starwalker']
place=['Barcelona','India', 'Germany', 'Venice', 'England']
where=['cinema', 'university','seminar', 'school', 'laundry']
occur=['made a lot of friends','Eats a burger', 'found a secret key', 'solved a mistery', 'wrote a book']
print(random.choice(when)+ ', '+random.choice(who)+' that lived in '+random.choice(place)+', went to the '+random.choice(where)+' and '+random.choice(occur))
```

Lesson 192

Full code

```
import random
cards=["Diamonds","Spades","Hearts","Clubs"]
nums=[2,3,4,5,6,7,8,9,10,"Jack","Queen","King","Ace"]
def pick_card():
    card=random.choice(cards)
    num=random.choice(nums)
    print('The ',num,' of ',card)

pick_card()
```

Explanation

Random card picker

Code

```
import random
cards=["Diamonds","Spades","Hearts","Clubs"]
nums=[2,3,4,5,6,7,8,9,10,"Jack","Queen","King","Ace"]
def pick_card():
```

Explanation

Define the function random().

Code

```
import random
cards=["Diamonds","Spades","Hearts","Clubs"]
nums=[2,3,4,5,6,7,8,9,10,"Jack","Queen","King","Ace"]
def pick_card():
    card=random.choice(cards)
    num=random.choice(nums)
```

Explanation

Make random choices.

Code

```
import random
cards=["Diamonds","Spades","Hearts","Clubs"]
nums=[2,3,4,5,6,7,8,9,10,"Jack","Queen","King","Ace"]
def pick_card():
    card=random.choice(cards)
    num=random.choice(nums)
    print('The ',num,' of ',card)

pick_card()
```

Explanation

Call the function and make random choices.

Your notes here

Debugging Lesson 193

Instruction:

Debuggign random library and choice

Code

```
import random
cards=["Diamonds","Spades","Hearts","Clubs"]
nums=[2,3,4,5,6,7,8,9,10,"Jack","Queen","King","Ace"]
def pick_card():
    card=random.choose(cards)
    num=random.choosee(nums)
    print('The ',num,' of ',card)

pick_card()
```

Lesson 194

Full code

```
def count_characters(s):
    count={}
    for i in s:
        if i in count:
            count[i]=count[i]+1
        else:
            count[i]=1
    print(count)
count_characters("Hi How are you")
```

Explanation

Count the occurrence of characters.

Code

```
def count_characters(s):
```

Explanation

s' is the parameter

Code

```
def count_characters(s):
    count={}
```

Explanation

Create empty dictionary.

Code

```
def count_characters(s):
    count={}
    for i in s:
```

Explanation

Loop through the parameter string.

Code

```
def count_characters(s):
    count={}
    for i in s:
        if i in count:
            count[i]=count[i]+1
        else:
            count[i]=1
```

Explanation

If the letter is present in dictionary then add 1 to it otherwise add as new key to the dictionary with count 1.

Your notes here

Debugging Lesson 195

Instruction:

Debugging functions and if statement

Code

```
def count_characters(s):
```

```
    count={}
    for i in s:
        if i in count:
            count[i]=count[i]+1
        else:
            count[i]=1
    print(count)

count_characters()
```

Lesson 196

Full code

```
def search(object,tosearch):
    found=False
    for i in object:
        if i==tosearch:
            found=True
            break
    return found

print(search(list(range(1,20)),15))
```

Explanation

Let's do a search

Code

```
def search(object,tosearch):
```

Explanation

Creates function which accepts the data and the number to search.

Code

```
def search(object,tosearch):
    found=False
```

Explanation

Set found variable to false

Code

```
def search(object,tosearch):
    found=False
    for i in object:
        if i==tosearch:
            found=True
```

Explanation

Loop through the data. If it's found then it's set to true.

Code

```
def search(object,tosearch):
    found=False
    for i in object:
        if i==tosearch:
            found=True
            break
    return found

print(search(list(range(1,20)),15))
```

Explanation

Call search function by passing data and number to search.

Your notes here

Debugging Lesson 197

Instruction:

Debugging functions and boolean

Code

```
def search(object,tosearch):
    found=False
    for i in object:
        if i==tosearch:
            found=true
```

```
            break
    Return found

print(search(list(range(1,20)),15))
```

Lesson 198

Full code

```
def anagram(word1,word2):
    word1=word1.lower()
    word2=word2.lower()
    return sorted(word1)==sorted(word2)

print(anagram("cinema","iceman"))
print(anagram("cool","loco"))
print(anagram("men","women"))
```

Explanation

Anagram means it will create another word when a word is rearranged.

Code

```
def anagram(word1,word2):
```

Explanation

Create function to accept two words.

Code

```
def anagram(word1,word2):
    word1=word1.lower()
    word2=word2.lower()
```

Explanation

Convert words to lower case.

Code

```
def anagram(word1,word2):
    word1=word1.lower()
    word2=word2.lower()
    return sorted(word1)==sorted(word2)
```

Explanation

If both words are same in sorted order, then it returns true.

Your notes here

Debugging Lesson 199

Instruction:

Debugging functions and sort

Code

```
def anagram(word1,word2):
    word1=word1.lower()
    word2=word2.lower()
    return sort(word1)==sort(word2)

print(anagram("cinema","iceman"))
print(anagram("cool","loco"))
print(anagram("men","women"))
```

Lesson 200

Full code

```
height=float(input('enter your height in centimeters:'))
weight=float(input('enter your weight in Kgs:'))
height=height/100
BMI=weight/(height*height)
print('your Body Mass Index is:',BMI)
```

Explanation

Calculate BMI

Code

```
height=float(input('enter your height in centimeters:'))
```

```
weight=float(input('enter your weight in Kgs:'))
```

Explanation

Get height and weight from user

Code

```
height=float(input('enter your height in centimeters:'))
weight=float(input('enter your weight in Kgs:'))
height=height/100
BMI=weight/(height*height)
```

Explanation

Calculate BMI using the formula

Your notes here

Debugging Lesson 201

Instruction:

Debugging print, float and input

Code

```
height=(input('enter your height in centimeters:'))
weight=float(input('enter your weight in Kgs:'))
height=height/100
BMI=weight/(height*height)

print('your Body Mass Index is:',BMI)
```

Lesson 202

Full code

```
height=float(input('enter your height in centimeters:'))
weight=float(input('enter your weight in Kgs:'))
height=height/100
BMI=weight/(height*height)
```

```
print('your Body Mass Index is:',BMI)
if(BMI>0):
    if(BMI<=16):
        print('you are severely underweight')
    elif(BMI<=18.5):
        print('you are underweight')
    elif(BMI<=25):
        print('you are Healthy')
    elif(BMI<=30):
        print('you are overweight')
    else: print('you are severely overweight')
else:('enter valid details')
```

Code

```
height=float(input('enter your height in centimeters:'))
weight=float(input('enter your weight in Kgs:'))
height=height/100
BMI=weight/(height*height)
print('your Body Mass Index is:',BMI)
if(BMI>0):
```

Explanation

BMI should be greater than 0

Code

```
height=float(input('enter your height in centimeters:'))
weight=float(input('enter your weight in Kgs:'))
height=height/100
BMI=weight/(height*height)
print('your Body Mass Index is:',BMI)
if(BMI>0):
    if(BMI<=16):
        print('you are severely underweight')
    elif(BMI<=18.5):
        print('you are underweight')
    elif(BMI<=25):
        print('you are Healthy')
    elif(BMI<=30):
        print('you are overweight')
```

Explanation

Based on BMI values, determine if a person is overweight or healthy.

Code

```
height=float(input('enter your height in centimeters:'))
weight=float(input('enter your weight in Kgs:'))
height=height/100
BMI=weight/(height*height)
print('your Body Mass Index is:',BMI)
if(BMI>0):
    if(BMI<=16):
        print('you are severely underweight')
    elif(BMI<=18.5):
        print('you are underweight')
    elif(BMI<=25):
        print('you are Healthy')
    elif(BMI<=30):
        print('you are overweight')
    else: print('you are severely overweight')
else:('enter valid details')
```

Explanation

If BMI is less than 0, then user has entered invalid details.

Your notes here

Debugging Lesson 203

Instruction:

Debugging if elif and else

Code

```
height=float(input('enter your height in centimeters:'))
weight=float(input('enter your weight in Kgs:'))
height=height/100
BMI=weight/(height*height)
print('your Body Mass Index is:',BMI)
if(BMI>0):
    if(BMI<=16):
```

```
        print('you are severely underweight')
    else if(BMI<=18.5):
        print('you are underweight')
    else if(BMI<=25):
        print('you are Healthy')
    else if(BMI<=30):
        print('you are overweight')
    else: print('you are severely overweight')
else:('enter valid details')
```

Lesson 204

Full code

```
while True:
    reply=input("Enter Text: ")
    if reply=='stop':
        break
    print(reply)
```

Explanation

Accept multiple inputs from user.

Code

```
while True:
```

Explanation

This condition will always be true, so the loop will never end.

Code

```
while True:
    reply=input("Enter Text: ")
    if reply=='stop':
        break
```

Explanation

If the user instructs it to stop, break the loop.

Your notes here

Debugging Lesson 205

Instruction:

Debugging while, boolean and break

Code

```
while True:
    reply=input("Enter Text: ")
    if reply === 'stop':
        break
    print(reply)
```

Lesson 206

Full code

```
product1_name,product1_price='Books', 50.95
product2_name,product2_price='Computer', 598.99
product3_name,product3_price='Monitor', 156.89
message='Thanks for shopping with us today!'
print('*'*60)
print('  Happy bookstore')
print('  144 MG Road')
print('  New Delhi')
print('='*40)
print(' Product Name  Product Price')
print(' {}      {}'.format(product1_name.title(), product1_price))
print(' {}    {}'.format(product2_name.title(), product2_price))
print(' {}     {}'.format(product3_name.title(), product3_price))
print('='*40)
print('   Total')
total = product1_price + product2_price + product3_price
print('         {}'.format(total))
print('='*40)
print(' Thanks for shopping with us today!
')
print('*'*60)
```

Explanation

Invoice generator

Code

```
product1_name,product1_price='Books', 50.95
product2_name,product2_price='Computer', 598.99
product3_name,product3_price='Monitor', 156.89
```

Explanation

Assign products and the price

Code

```
product1_name,product1_price='Books', 50.95
product2_name,product2_price='Computer', 598.99
product3_name,product3_price='Monitor', 156.89
message='Thanks for shopping with us today!'
print('*'*60)
print('  Happy bookstore')
```

Explanation

Start printing the bill

Code

```
product1_name,product1_price='Books', 50.95
product2_name,product2_price='Computer', 598.99
product3_name,product3_price='Monitor', 156.89
message='Thanks for shopping with us today!'
print('*'*60)
print('  Happy bookstore')
print('
```

Explanation

Will include a tab space in the bill.

Code

```
product1_name,product1_price='Books', 50.95
product2_name,product2_price='Computer', 598.99
product3_name,product3_price='Monitor', 156.89
message='Thanks for shopping with us today!'
print('*'*60)
```

```
print('  Happy bookstore')
print('  144 MG Road')
print('  New Delhi')
print('='*40)
print(' Product Name  Product Price')
print(' {}      {}'.format(product1_name.title(), product1_price))
print(' {}    {}'.format(product2_name.title(), product2_price))
print(' {}     {}'.format(product3_name.title(), product3_price))
print('='*40)
print('   Total')
total = product1_price + product2_price + product3_price
print('         {}'.format(total))
```

Explanation

Format will replace with values.

Your notes here

Debugging Lesson 207

Instruction:

Debugging tab space and new line, print

Code

```
product1_name,product1_price='Books', 50.95
product2_name,product2_price='Computer', 598.99
product3_name,product3_price='Monitor', 156.89

message='Thanks for shopping with us today!'

print('*'*60)

print('  Happy bookstore')
print('  144 MG Road')
print('  New Delhi')
```

```
print('='*40)

print(' Product Name  Product Price')

print(' {}     {}'.f(product1_name.title(), product1_price))
print(' {}   {}'.f(product2_name.title(), product2_price))
print(' {}    {}'.f(product3_name.title(), product3_price))

print('='*40)

print('   Total')
total = product1_price + product2_price + product3_price

print('        {}'.f(total))

print('='*40)

print(' Thanks for shopping with us today!
')

print('*'*60)
```

Lesson 208

Full code

```
list=['a','b','v','g']
for i in reversed(list):
    print(i)
```

Explanation

Backward for loop.

Code

```
list=['a','b','v','g']
for i in reversed(list):
```

Explanation

reversed() function is used to reverse the elements of a sequence.

Your notes here

Debugging Lesson 209

Instruction:

Debugging reversed and Syntax

Code

```
list=['a','b','v','g']
for i in reverse(list):
    print(i)
```

Lesson 210

Full code

```
word='mom'
word=word.lower()
if word==word[::-1]:
    print(word,' is a palindrome')
else:
    print(word,' is not a palindrome')
```

Explanation

Palindrome words

Code

```
word='mom'
```

Explanation

Palindrome is a word that reads the same backward or forward

Code

```
word='mom'
word=word.lower()
```

Explanation

Convert word to lower case

Code

```
word='mom'
word=word.lower()
if word==word[::-1]:
```

Explanation

Check to see if the word and it's reverse are the same.

Code

```
word='mom'
word=word.lower()
if word==word[:
```

Explanation

Indexing is used.The word is reversed and compared to the original word.

Your notes here

Debugging Lesson 211

Instruction:

Debugging lower and Syntax

Code

```
word='mom'
word=word.lowercase()
if word==word[::-1]:
    print(word,' is a palindrome')
else:
    print(word,' is not a palindrome')
```

Lesson 212

Full code

```
def palindrome(sentence):
    for i in (",.'?/><}{{}}'"):
        sentence=sentence.replace(i, "")
    palindrome=[]
    words=sentence.split(' ')
    for word in words:
        word=word.lower()
        if word==word[::-1]:
            palindrome.append(word)
    return palindrome

sentence=input("Enter a sentence : ")
print(palindrome(sentence))
```

Explanation

Find palindrome words in a sentence Ex: Mom, noon, peep, wow are palindrome words.

Code

```
def palindrome(sentence):
```

Explanation

Create a function to look for palindromes in words.

Code

```
def palindrome(sentence):
    for i in (",.'?/><}{{}}'"):
        sentence=sentence.replace(i, "")
```

Explanation

Replace all special characters in sentence with spaces.

Code

```
def palindrome(sentence):
    for i in (",.'?/><}{{}}'"):
        sentence=sentence.replace(i, "")
    palindrome=[]
    words=sentence.split(' ')
```

Explanation

split() method forms a list of words.

Your notes here

Debugging Lesson 213

Instruction:

Debugging functions and Syntax

Code

```
def palindrome():
    for i in (",.'?/><}{{}}'"):
        sentence=sentence.replace(i, "")
    palindrome=[]
    words=sentence.split(' ')
    for word in words:
        word=word.lower()
        if word==word[::-1]:
            palindrome.append(word)
    return palindrome

sentence=input("Enter a sentence : ")
print(palindrome(sentence))
```

Lesson 214

Full code

```
def maximum(x):
    max_index=0
    cur_index=1
    while cur_indexx[max_index]:
            max_index=cur_index
        cur_index=cur_index+1
    return max_index
a=[23, 76, 45, 20, 70, 65, 15, 54]
print(maximum(a))
```

Explanation

Find index of maximum value

Code

```
def maximum(x):
    max_index=0
    cur_index=1
```

Explanation

Initialize 2 variables.

Code

```
def maximum(x):
    max_index=0
    cur_index=1
    while cur_index<len(x):
```

Explanation

Continue the loop till cur_index is less than length of the list.

Code

```
def maximum(x):
    max_index=0
    cur_index=1
    while cur_indexx[max_index]:
```

Explanation

Take 2 values and check which is greater.

Code

```
def maximum(x):
    max_index=0
    cur_index=1
    while cur_indexx[max_index]:
            max_index=cur_index
```

Explanation

Whichever is greater, store the index value in max_index.

Your notes here

Debugging Lesson 215

Instruction:

Debugging functions, while and indexing

Code

```
def maximum(x):
    max_index=0
    cur_index=1
    while cur_indexx(max_index):
            max_index=cur_index
        cur_index=cur_index+1
    return max_index
a=[23, 76, 45, 20, 70, 65, 15, 54]
print(maximum(a))
```

Lesson 216

Full code

```
def minimum(x):
    min_index=0
    cur_index=1
    while cur_index<len(x):
        if x[cur_index]<x[min_index]:
            min_index=cur_index
        cur_index=cur_index+1
    return min_index
a=[23, 76, 45, 20, 70, 65, 15, 54]
print(minimum(a))
```

Explanation

Find index of minimum value

Code

```
def minimum(x):
    min_index=0
    cur_index=1
    while cur_index<len(x):
```

Explanation

Loop through till current index is less than length of the list.

Code

```
def minimum(x):
    min_index=0
    cur_index=1
    while cur_index<len(x):
        if x[cur_index]<x[min_index]:
```

Explanation

Compare two numbers and get the minimum number.

Code

```
def minimum(x):
    min_index=0
    cur_index=1
    while cur_index<len(x):
        if x[cur_index]<x[min_index]:
            min_index=cur_index
```

Explanation

min_index will always store the index of minimum value. So, if the current index value is less than the minimum index value, then the current index value will be stored in min_index.

Your notes here

Debugging Lesson 217

Instruction:

Debugging functions, while and indexing

Code

```
defining minimum(x):
    min_index=0
    cur_index=1
    while cur_index<len(x):
        if x[cur_index]<x[min_index]:
            min_index=cur_index
        cur_index=cur_index+1
    return min_index

a=[23, 76, 45, 20, 70, 65, 15, 54]
print(minimum(a))
```

Lesson 218

Full code

```
my_str="Hello this Is an Example With cased letters"
sent=my_str.split()
words=[]
for word in sent:
    words.append(word.lower())
words.sort()
print("The sorted words are:",words)
```

Explanation

Sort words in a sentence

Code

```
my_str="Hello this Is an Example With cased letters"
sent=my_str.split()
```

Explanation

Split the sentence and form a list.

Code

```
my_str="Hello this Is an Example With cased letters"
sent=my_str.split()
words=[]
```

Explanation

Create an empty list to store the sorted words.

Code

```
my_str="Hello this Is an Example With cased letters"
sent=my_str.split()
words=[]
for word in sent:
    words.append(word.lower())
```

Explanation

Read through the list and convert it to lower case.

Code

```
my_str="Hello this Is an Example With cased letters"
sent=my_str.split()
words=[]
for word in sent:
    words.append(word.lower())
words.sort()
```

Explanation

Next, perform a sorting operation.

Your notes here

Debugging Lesson 219

Instruction:

Debugging string, split and for statement,append

Code

```
my_str="Hello this Is an Example With cased letters"
sent=split(my_str)
words=[]
for word in sent:
```

```
    words.append(word.lower())
words.sort()
print("The sorted words are:",words)
```

Lesson 220

Full code

```
for i in range(1,100):
    if i%7==0:
        print(i)
```

Explanation

Program to find the numbers that are divisible by 7.

Code

```
for i in range(1,100):
```

Explanation

Find the numbers divisible by 7 between 1 and 100.

Code

```
for i in range(1,100):
    if i%7==0:
```

Explanation

Divide the given number by 7 and if the reminder is 0, then the number is divisible by 7.

Your notes here

Debugging Lesson 221

Instruction:

Debugging for, range and modulo

Code

```
for i in range(1,100):
    if i%7= 0:
        print(i)
```

Lesson 222

Full code

```
num=75869
count=0
while num!=0:
    num=num//10
    count+= 1
print("Total digits are: ", count)
```

Explanation

Count the total number of digits in a number

Code

```
num=75869
```

Explanation

num is the number to count number of digits

Code

```
num=75869
count=0
while num!=0:
```

Explanation

Continue loop till it's not 0.

Code

```
num=75869
count=0
while num!=0:
    num=num//10
```

Explanation

Divide number by 10 and get the nearest whole number

Code

```
num=75869
count=0
while num!=0:
    num=num//10
    count+= 1
```

Explanation

count will keep counting the digits

Your notes here

Debugging Lesson 223

Instruction:

Debugging while and floor division

Code

```
num=75869
count=0
while num!==0:
    num=num//10
    count+= 1

print("Total digits are: ", count)
```

Lesson 224

Full code

```
def sum(numbers):
    total=0
    for x in numbers:
        total+=x
    return total
print(sum((8, 2, 3, 0, 7)))
```

Explanation

Calculate sum of numbers

Code

```
def sum(numbers):
    total=0
```

Explanation

Initialize a variable name total.

Code

```
def sum(numbers):
    total=0
    for x in numbers:
```

Explanation

Read through the numbers.

Code

```
def sum(numbers):
    total=0
    for x in numbers:
        total+=x
```

Explanation

This is equivalent to total=total+x.

Code

```
def sum(numbers):
    total=0
    for x in numbers:
        total+=x
    return total
```

Explanation

Numbers will be summed up one by one.

Your notes here

Debugging Lesson 225

Instruction:

Debugging functions and for

Code

```
def sum(numbers)
    total=0
    for x is numbers:
        total+=x
    return total

print(sum((8, 2, 3, 0, 7)))
```

Lesson 226

Full code

```
def string_count(str1):
    d={'upper':0, 'lower':0}
    for i in str1:
        if i.isupper():
            d['upper']=d['upper']+1
        elif i.islower():
            d['lower']=d['lower']+1
        else:
            pass
    print('Upper case characters :',d['upper'])
    print('Lower case Characters :',d['lower'])

string_count('Jack and Jill went up the Hill')
```

Explanation

Count number of upper and lower case characters in a string.

Code

```
def string_count(str1):
    d={'upper':0, 'lower':0}
```

Explanation

Create a dictionary to store results.

Code

```
def string_count(str1):
    d={'upper':0, 'lower':0}
    for i in str1:
```

Explanation

Read through the string.

Code

```
def string_count(str1):
    d={'upper':0, 'lower':0}
    for i in str1:
        if i.isupper():
            d['upper']=d['upper']+1
```

Explanation

If character is in upper case , add to upper key.

Code

```
def string_count(str1):
    d={'upper':0, 'lower':0}
    for i in str1:
        if i.isupper():
            d['upper']=d['upper']+1
        elif i.islower():
            d['lower']=d['lower']+1
```

Explanation

If character is in lower case , add to lower key.

Code

```
def string_count(str1):
    d={'upper':0, 'lower':0}
    for i in str1:
        if i.isupper():
            d['upper']=d['upper']+1
```

```
        elif i.islower():
            d['lower']=d['lower']+1
        else:
            pass
```

Explanation

The pass does nothing. It is used when a statement is required syntactically but you do not want any command or code to execute.

Your notes here

Debugging Lesson 227

Instruction:

Debugging function, if elif else

Code

```
def string_count(str1):
    d={'upper':0, 'lower':0}
    for i in str1:
        if i.isUpper():
            d['upper']=d['upper']+1
        elif i.isLower():
            d['lower']=d['lower']+1
        else:

    print('Upper case characters :',d['upper'])
    print('Lower case Characters :',d['lower'])

string_count('Jack and Jill went up the Hill')
```

Lesson 228

Full code

```
nums=(1, 2, 3, 4, 5, 6, 7)
print("Original list:",nums)
```

```
triplelist=[]
for j in nums:
    triplelist.append(j+j+j)
print("
Triple of said list numbers:")
print(triplelist)
```

Explanation

Triple all numbers in a list.

Code

```
nums=(1, 2, 3, 4, 5, 6, 7)
print("Original list:",nums)
triplelist=[]
```

Explanation

Create an empty list

Code

```
nums=(1, 2, 3, 4, 5, 6, 7)
print("Original list:",nums)
triplelist=[]
for j in nums:
    triplelist.append(j+j+j)
```

Explanation

Triple the number and add it to empty list.

Your notes here

Debugging Lesson 229

Instruction:

Debugging for, list and append

Code

```
nums=(1, 2, 3, 4, 5, 6, 7)
print("Original list:",nums)
triplelist=[]
for j in nums:
    triplelist.add(j+j+j)

print("
Triple of said list numbers:")
print(triplelist)
```

Lesson 230

Full code

```
def change_cases(s):
    res=[]
    for i in s:
        if i.isupper():
            res.append(i.lower())
        elif i.islower():
            res.append(i.upper())
    return res

characters={'a', 'b', 'E', 'f', 'a', 'i', 'o', 'U', 'a'}
print("Original Characters:
",characters)
result = change_cases(characters)
print("After converting above characters in upper and lower cases and
eliminating duplicate letters:")
print(set(result))
```

Explanation

Conversion of upper and lower cases

Code

```
def change_cases(s):
    res=[]
    for i in s:
        if i.isupper():
            res.append(i.lower())
        elif i.islower():
```

```
            res.append(i.upper())
```

Explanation

Fetch each element and check the case, if its lower, convert to upper case and if its upper, convert to lower case.

Code

```
def change_cases(s):
    res=[]
    for i in s:
        if i.isupper():
            res.append(i.lower())
        elif i.islower():
            res.append(i.upper())
    return res
```

Explanation

After conversion, add it to the list.

Code

```
def change_cases(s):
    res=[]
    for i in s:
        if i.isupper():
            res.append(i.lower())
        elif i.islower():
            res.append(i.upper())
    return res

characters={'a', 'b', 'E', 'f', 'a', 'i', 'o', 'U', 'a'}
print("Original Characters:
",characters)
result = change_cases(characters)
print("After converting above characters in upper and lower cases and
eliminating duplicate letters:")
print(set(result))
```

Explanation

Converting it to set will eliminate duplicates.

Your notes here

Debugging Lesson 231

Instruction:

Debugging functions, for and str function

Code

```
def change_cases(s):
    res={}
    for i in s:
        if i.isupper():
            res.append(i.lower())
        elif i.islower():
            res.append(i.upper())
    return res

characters={'a', 'b', 'E', 'f', 'a', 'i', 'o', 'U', 'a'}
print(""Original Characters:
"",characters)

result = change_cases(characters)
print(""
After converting above characters in upper and lower cases
and eliminating duplicate letters:"")
print(set(result))
```

Lesson 232

Full code

```
def multiply(numbers):
    total=1
    for x in numbers:
        total*=x
    return total
print(multiply((8, 2, 3, -1, 7)))
```

Explanation

Find product of all numbers in a list

Code

```
def multiply(numbers):
    total=1
    for x in numbers:
```

Explanation

Define function to accept list and loop through it.Initialize a variable name total.

Code

```
def multiply(numbers):
    total=1
    for x in numbers:
        total*=x
```

Explanation

*This is equivalent to total=total*x.Multiply the current number to the total.*

Your notes here

Debugging Lesson 233

Instruction:

Debugging function and for

Code

```
def multiply(numbers):
    total=1
    for x in numbers:
        total*=x
    return total
print(multiply[8, 2, 3, -1, 7])
```

Lesson 234

Full code

```
def reverse_word(str1):
    return ' '.join(reversed(str1.split()))
print(reverse_word('Welcome to python'))
```

Explanation

Reverse a string word by word.

Code

```
def reverse_word(str1):
    return ' '.join(reversed(str1.split()))
```

Explanation

str1.split() will create a list and the list will be reversed and join will convert the reversed list to string.

Your notes here

Debugging Lesson 235

Instruction:

Debugging function, reverse and join

Code

```
def reverse_word(str1):
    return ' '.joined(reversed(str1.split()))
print(reverse_word('Welcome to python'))
```

Lesson 236

Full code

```
def sumofcube(n):
    sum=0
```

```
    for i in range(1,n+1):
        sum=sum+(i*i*i)
    return sum
print(sumofcube(5))
```

Explanation

Find sum of cubes of n natural numbers.

Code

```
def sumofcube(n):
    sum=0
```

Explanation

Initialize a variable sum.

Code

```
def sumofcube(n):
    sum=0
    for i in range(1,n+1):
```

Explanation

Generate numbers from 1 to n using range() function.

Code

```
def sumofcube(n):
    sum=0
    for i in range(1,n+1):
        sum=sum+(i*i*i)
```

Explanation

Multiplying the i value three times and adding to the sum.

Your notes here

Lesson 238

Full code

```
import turtle
def draw_head(doll):
    doll.circle(60)
    doll.right(60)
def draw_body(doll):
    for i in range(3):
        doll.forward(150)
        doll.right(120)

t = turtle.Turtle()
t.color("pink")

draw_head(t)
draw_body(t)
```

Explanation

Lets draw a doll step by step.

Code

```
import turtle
def draw_head(doll):
    doll.circle(60)
    doll.right(60)
```

Explanation

Draw head using circle

Code

```
import turtle
def draw_head(doll):
    doll.circle(60)
    doll.right(60)
def draw_body(doll):
    for i in range(3):
        doll.forward(150)
        doll.right(120)
```

Explanation

Draw body with a triangle and it has 3 lines so loop it 3 times.

Your notes here

Lesson 240

Full code

```
import turtle
def draw_head(doll):
    doll.circle(60)
    doll.right(60)
def draw_body(doll):
    for i in range(3):
        doll.forward(150)
        doll.right(120)
def draw_arm(doll):
    doll.forward(60)
    doll.left(60)

    doll.forward(60)
    doll.backward(60)

    doll.right(60)
    doll.backward(60)

    doll.right(60)

    doll.forward(60)
    doll.right(60)

    doll.forward(60)
    doll.backward(60)

    doll.left(60)
    doll.forward(90)

def draw_legs(doll):
    doll.left(120)
```

```
    doll.forward(40)
    doll.right(120)
    doll.forward(120)
    doll.right(180)
    doll.forward(120)
    doll.right(60)
    doll.forward(70)
    doll.right(60)
    doll.forward(120)
t=turtle.Turtle()
t.color("red")

draw_head(t)
draw_body(t)
draw_arm(t)
draw_legs(t)
```

Code

```
import turtle
def draw_head(doll):
    doll.circle(60)
    doll.right(60)
def draw_body(doll):
    for i in range(3):
        doll.forward(150)
        doll.right(120)
```

Explanation

Using commands forward , backward, left and right, move the cursor and draw the line.

Code

```
import turtle
def draw_head(doll):
    doll.circle(60)
    doll.right(60)
def draw_body(doll):
    for i in range(3):
        doll.forward(150)
        doll.right(120)
def draw_arm(doll):
```

Explanation

draw arms on two sides of a triangle.

Code

```
import turtle
def draw_head(doll):
    doll.circle(60)
    doll.right(60)
def draw_body(doll):
    for i in range(3):
        doll.forward(150)
        doll.right(120)
def draw_arm(doll):
    doll.forward(60)
    doll.left(60)

    doll.forward(60)
    doll.backward(60)

    doll.right(60)
    doll.backward(60)

    doll.right(60)

    doll.forward(60)
    doll.right(60)

    doll.forward(60)
    doll.backward(60)

    doll.left(60)
    doll.forward(90)

def draw_legs(doll):
```

Explanation

Draw legs at the bottom of the triangle.

Your notes here

Lesson 242

Full code

```
import turtle
import random

wn=turtle.Screen()
wn.bgcolor("orange")

p1=turtle.Turtle()
p1.color('red')
p1.shape('turtle')
p1.penup()
p1.goto(-120,80)
p1.pendown()

for i in range(100):
    p1.forward(random.randint(1,5))
```

Code

```
import turtle
import random

wn=turtle.Screen()
wn.bgcolor("orange")
```

Explanation

Set background color to orange.

Code

```
import turtle
import random

wn=turtle.Screen()
wn.bgcolor("orange")

p1=turtle.Turtle()
p1.color('red')
p1.shape('turtle')
```

Explanation

Set shape and colour of cursor.

Code

```
import turtle
import random

wn=turtle.Screen()
wn.bgcolor("orange")

p1=turtle.Turtle()
p1.color('red')
p1.shape('turtle')
p1.penup()
p1.goto(-120,80)
```

Explanation

Set cursor position.

Code

```
import turtle
import random

wn=turtle.Screen()
wn.bgcolor("orange")

p1=turtle.Turtle()
p1.color('red')
p1.shape('turtle')
p1.penup()
p1.goto(-120,80)
p1.pendown()

for i in range(100):
    p1.forward(random.randint(1,5))
```

Explanation

Draw the line

Your notes here